Lecture 3
Retrieval Models
Part 2

This Part Composed of Contents Come from Courseware of Below Professors:
James Allan, University of Massachusetts Amherst
Gerald Benoit, Simmons College
Pandu Nayak and Prabhakar Raghavan, Stanford University
Edited by: Qingcai Chen, HIT Shenzhen Graduate School

Models we'll consider

- *Boolean (exact match)*
- *Statistical language models*
- Vector space
- Latent Semantic Indexing
- Inference network (推理网络)
- Classic probabilistic approaches

this lecture

Left to latter lectures

Document Representation in Boolean Model

What's the main issue of this representation?

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

1 if play contains word, 0 otherwise

Brutus *AND* ***Caesar*** *BUT NOT* ***Calpurnia***

Questions before the VSM

- How to measure the distance of two complicate objects in machine learning?
- How to mathematically measure the relevance of two objects?
- Why we think that Boolean model is not a good model for text retrieval?

Vector Space Model

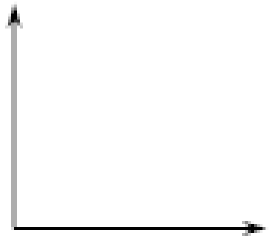
- Variations (不同形式):
  - Vector space retrieval model (向量空间模型)
  - Latent Semantic Indexing (潜层语义索引)
- Key idea:
  - Everything (documents, queries, terms) is a vector in a high-dimensional space
- Example systems
  - SMART,
    - G. Salton and students at Cornell starting in the 60’s
  - Lucene
    - popular open source search engine written in Java,
    - still be a building block of many commercial SEs
  - Most Web search engines are similar

Vector space issues

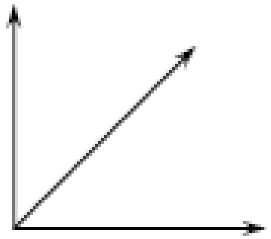
- How to select basis vectors(基向量) (dimensions)
- How to convert objects into vectors
  - Terms
  - Documents
  - Queries
- How to select magnitude (幅值) along a dimension
- How to compare objects in vector space
  - Comparing queries to documents

Vector Space and Basis Vectors

- Formally, a *vector space* is defined by a set of *linearly independent* (线性独立) basis vectors. (Why?)
- Basis vectors:
  - correspond to the *dimensions* or *directions* in the vector space;
  - determine what can be described in the vector space; and
  - must be *orthogonal* (正交), or *linearly independent,* i.e. a value along one dimension implies nothing about a value along another.



Basis vectors for 2 dimensions



Basis vectors for 3 dimensions

Selection of Basic Vector

- What should be the basis vectors for IR? (feature selection problem)
- "Core" concepts of discourse?*
  - orthogonal (by definition)
  - a relatively static vector space
  - probably not too many dimensions
  - ***But…*** difficult to determine (Philosophy? Cognitive science?)
- Use terms that appear?
  - easy to determine
  - ***But…***
    - not at all orthogonal (but it may not matter much)
    - a constantly growing vector space (new vocabulary)
    - huge number of dimensions

Selection of Basic Vector

HowNet(董振东、董强）的语义表示与计算

├{ActGeneral|泛动} {act|行动:agent={*}}
│ ├{start|开始} {ActGeneral|泛动:agent={*},content={*}}
│ ├{do|做} {ActGeneral|泛动:agent={*},content={*},manner={*}}
│ │ ├{try|尝试} {do|做:agent={*},content={*}}

ience?)

- Use terms
  - easy to
  - ***But…***
    - not at
    - a con
    - huge

employer:	DEF=human|人, *employ|雇用
employee:	DEF= human|人, $employ|雇用
iron:	DEF=tool|用具, *AlterForm|变形状, #level|平
vacation:	DEF=time|时间, @rest|休息, @WhileAway|消闲
hotel:	DEF=InstitutePlace|场所, @reside|住下,#tour|旅游
lifeboat:	DEF=ship|船,*rescue|救助

Selection of Basic Vector

- What should be the basis vectors for IR? (feature selection problem)
- “Core” concepts of discourse?*
  - orthogonal (by definition)
  - a relatively static vector space
  - probably not too many dimensions
  - ***But…*** difficult to determine (Philosophy? Cognitive science?)
- Use terms that appear?
  - easy to determine
  - ***But…***
    - not at all orthogonal (but it may not matter much)
    - a constantly growing vector space (new vocabulary)
    - huge number of dimensions

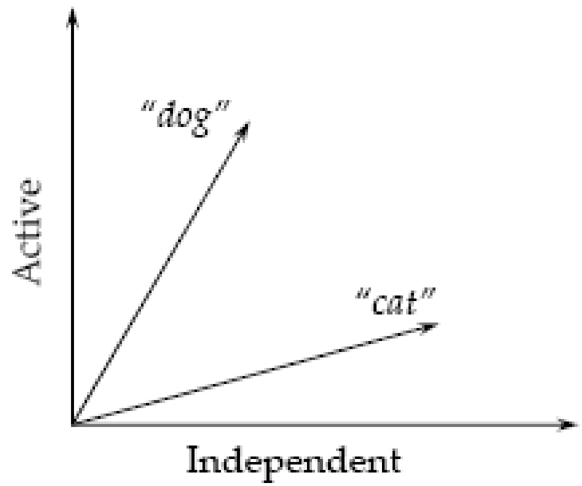
Mapping to basis vectors: terms

- How do basis vectors relate to terms?
  - Each term is represented as a linear combination of basis vectors.

Dictionary

"Basis terms"

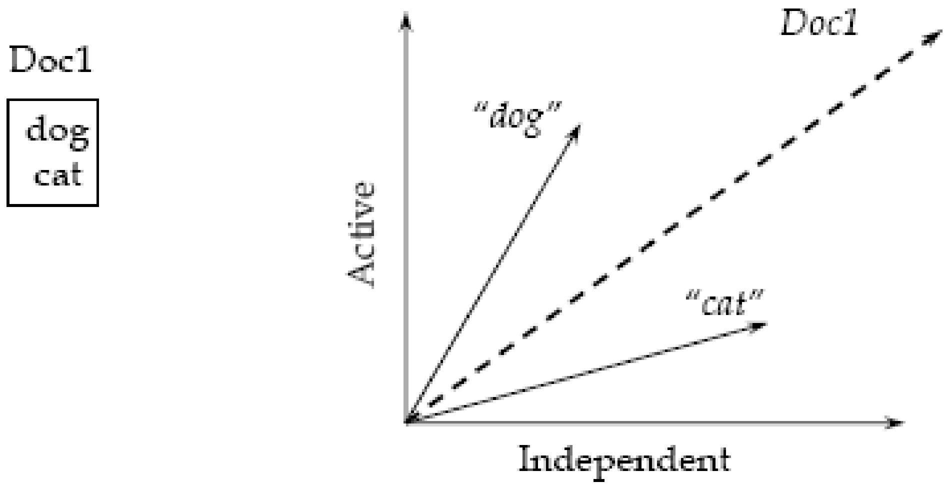
Term	Active	Independent
cat	0.25	0.75
dog	0.75	0.25
gravity	?	?



cat = 0.25 *Active* + 0.75 *Independent* (**or** cat = 0.25 x + 0.75 y)
dog = 0.75 x + 0.25 y

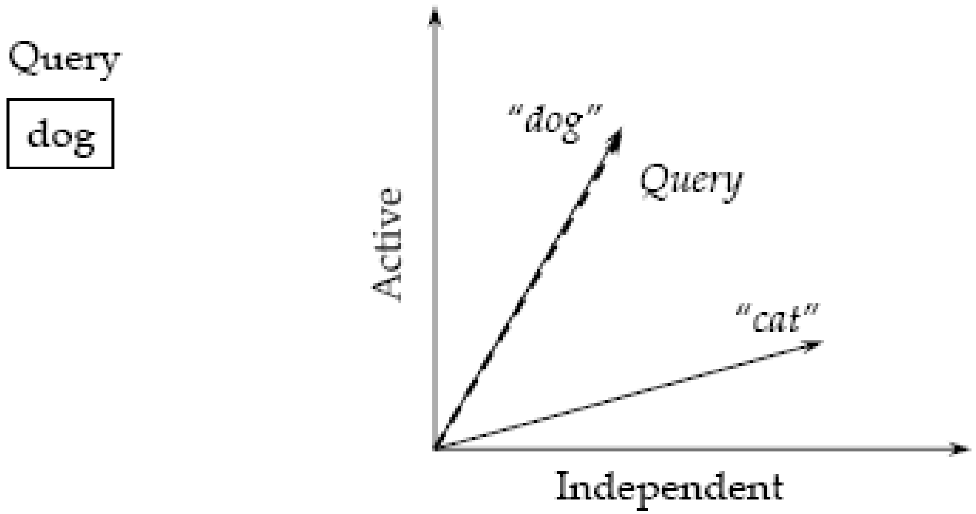
Mapping to basis vectors: documents

- How are documents represented?
  - A document is represented as the sum of its term vectors.



Mapping to basis vectors: queries

- How are queries represented?
  - Same way that documents are



Vector Coefficients

- The coefficients (vector lengths, term weights) represent term presence, importance, or “aboutness”
  - Magnitude along each dimension
- Model gives no guidance on how to set term weights
- Some common choices:
  - Binary: 1 = term is present, 0 = term not present in document
  - *tf*: The frequency of the term in the document
  - *tf idf (inverse document frequency)* indicates the discriminatory power (辨识能力) of the term (why?)

- Tf·idf is far from the most common
  - Numerous variations…

Term weighting functions (e.g.)

- Lucene weighting function

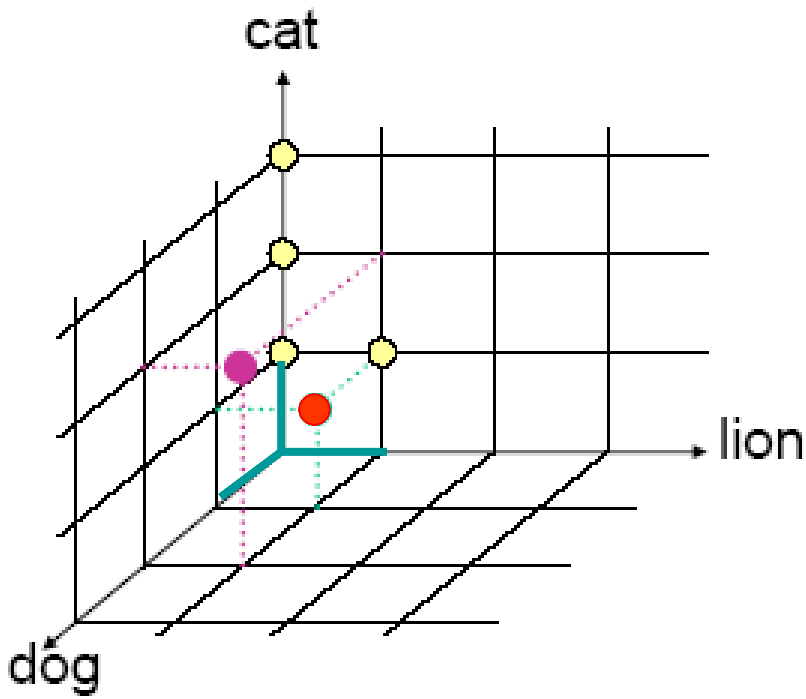
$$w_{t,d} = \frac{\mathrm{tf}_{d,t} \cdot \log(N/\mathrm{df}_t + 1)}{\sqrt{\text{number of tokens in } d \text{ in the same field as } t}}$$

- Smart supports a number of functions, XYZ
  - X expresses term frequency component
  - Y expressed inverse document frequency component
  - Z expresses (length) normalization component
  - e.g., atc = augmented tfidf cosine

$$\frac{\left(\frac{1}{2} + \frac{1}{2}\frac{\mathrm{tf}_{t,d}}{\max(\mathrm{tf}_{*,d})}\right) \cdot \log \frac{N}{n_t}}{\left[\sum_t \left(\left(\frac{1}{2} + \frac{1}{2}\frac{\mathrm{tf}_{t,d}}{\max(\mathrm{tf}_{*,d})}\right) \cdot \log \frac{N}{n_t}\right)^2\right]^{0.5}}$$

Example: 3-word vocabulary (tf weights)

- cat
- cat cat
- cat cat cat
- cat lion
- lion cat
- cat lion dog
- cat cat lion dog dog



Similarity

Problem: Given two text documents, how **similar** are they?

[Methods that measure similarity do not assume exact matches.]

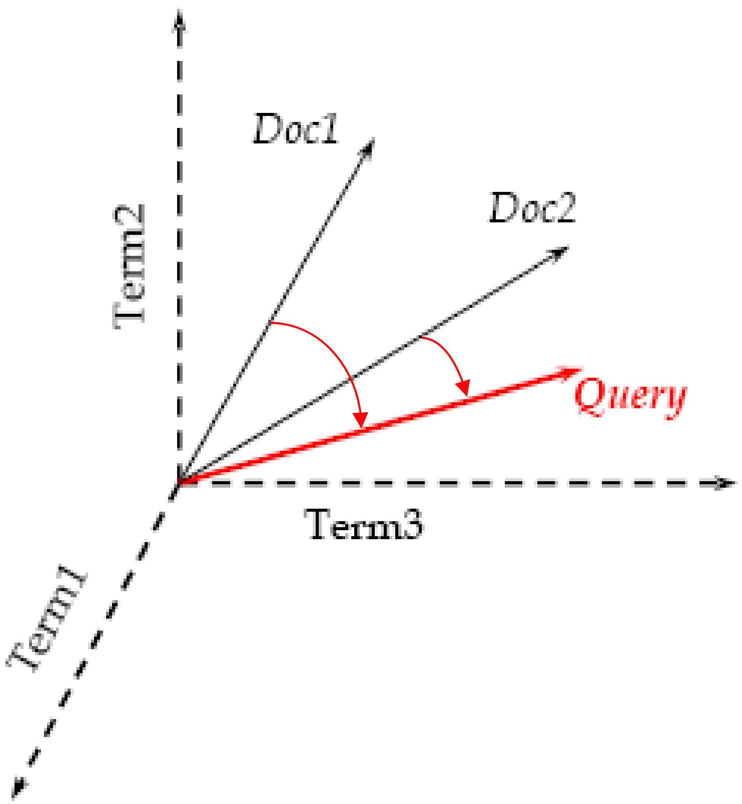
Example

Here are three documents. How similar are they?

d_1 *ant ant bee*
d_2 *dog bee dog hog dog ant dog*
d_3 *cat gnu dog eel fox*

Documents can be any length from one word to thousands.
A query is a special type of document.

Vector Space Similarity



Similarity is inversely related to the angle between the vectors.

Doc2 is the most similar to the *Query*.

Rank the documents by their similarity to the *Query*.

Vector Space Similarity: Weighted Features Example

$T_1 \quad T_2 \quad T_3$

$D_1 = 3\ cat + 1\ dog + 4\ lion$
$D_2 = 8\ cat + 2\ dog + 6\ lion$

$Q = 2\ dog$

$D_1 = (3T_1 + 1T_2 + 4T_3)$
$D_2 = (8T_2 + 2T_2 + 6T_3)$

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

Correlated Terms

	Term	cat	dog	lion
T_1	cat	1.00	-0.20	0.50
T_2	dog	-0.20	1.00	-0.40
T_3	lion	0.50	-0.40	1.00

Orthogonal Terms

Term	cat	dog	lion
cat	1.00	0.00	0.00
dog	0.00	1.00	0.00
lion	0.00	0.00	1.00

$$
\begin{aligned}
Sim(D_1,Q) &= (3T_1 + 1T_2 + 4T_3) \bullet (2T_2) \\
&= 6T_1 \bullet T_2 + 2T_2 \bullet T_2 + 8T_3 \bullet T_2 \\
&= -6 \bullet 0.2 + 2 \bullet 1 - 8 \bullet 0.4 \\
&= -1.2 + 2 - 3.2 \\
&= -2.4
\end{aligned}
$$

$$
\begin{aligned}
Sim(D_1,Q) &= 3 \bullet 0 + 1 \bullet 2 + 4 \bullet 0 \\
&= 2
\end{aligned}
$$

词语的One-hot表示？

Vector Space Similarity: Common Measures

Sim(X,Y)	Binary Term Vectors	Weighted Term Vectors
Inner product	$\lvert X \cap Y \rvert$	$\sum x_i . y_i$
Dice coefficient	$\frac{2 \lvert X \cap Y \rvert}{\lvert X \rvert + \lvert Y \rvert}$	$\frac{2 \sum x_i . y_i}{\sum x_i^2 + \sum y_i^2}$
Cosine coefficient	$\frac{\lvert X \cap Y \rvert}{\sqrt{\lvert X \rvert} \sqrt{\lvert Y \rvert}}$	$\frac{\sum x_i . y_i}{\sqrt{\sum x_i^2 . \sum y_i^2}}$
Jaccard coefficient	$\frac{\lvert X \cap Y \rvert}{\lvert X \rvert + \lvert Y \rvert - \lvert X \cap Y \rvert}$	$\frac{\sum x_i . y_i}{\sum x_i^2 + \sum y_i^2 - \sum x_i . y_i}$

Vector Space Similarity: Cosine Coefficient (Correlation) Example

$D_1 = (0.5T_1 + 0.8T_2 + 0.3T_3)$ $\qquad$ $Q = (1.5T_1 + 1T_2 + 0T_3)$

$$\mathrm{Sim}(D_1,Q) = \frac{(0.5\times 1.5)+(0.8\times 1)}{\sqrt{(0.5^2+0.8^2+0.3^2)(1.5^2+1^2)}}$$

$$= \frac{1.55}{\sqrt{.98\times 3.25}}$$

$$= .868$$

Cosine and vector lengths

- Angle is independent of vector lengths
- Can normal all vectors to length one

$$\frac{\sum x_i \cdot y_i}{\sqrt{\sum x_i^2 \cdot \sum y_i^2}} \Rightarrow \sum x_i \cdot y_i$$

- Inner product equals cosine of angle
- Inner product more efficient to compute
- Very common to normalize vector lengths in index

Example again, normalized

$$D_1 = (0.5T_1 + 0.8T_2 + 0.3T_3) \qquad Q = (1.5T_1 + 1T_2 + 0T_3)$$

$$\begin{aligned} D'_1 &= (0.5T_1 + 0.8T_2 + 0.3T_3)/\sqrt{0.98} \\ &\approx 0.51T_1 + 0.82T_2 + 0.31T_3 \end{aligned} \qquad \begin{aligned} Q' &= (1.5T_1 + 1T_2 + 0T_3)/\sqrt{3.25} \\ &\approx 0.83T_1 + 0.555T_2 \end{aligned}$$

$$\begin{aligned} \text{Sim}(D_1, Q) &= \text{Sim}(D'_1, Q') \\ &= \frac{(0.51 \times 0.83) + (0.82 \times 0.555)}{\sqrt{(0.51^2 + 0.82^2 + 0.31^2)(0.83^2 + 0.555^2)}} \\ &= (0.51 \times 0.83) + (0.82 \times 0.555) \\ &= 0.878 \\ &\approx 0.868 \text{ (from earlier slide)} \end{aligned}$$

Other comparisons: Lucene

tf·idf from document

User-specified boost

$$\sum_t \left(\frac{\mathrm{tf}_{q,t} \cdot \mathrm{idf}_t}{\mathrm{norm}_q} \frac{\mathrm{tf}_{d,t} \cdot \mathrm{idf}_t}{\mathrm{norm}_{d,t}} \mathrm{boost}_t \right) \cdot \frac{\mathrm{overlap}(q,d)}{|q|}$$

Length-normalized query weight

Term normalization is square root of number of tokens in *d* that are in the same field as *t*

Proportion of query matched

Summary: Vector Similarity Computation with Weights

Documents in a collection are assigned **terms** from a set of n terms

The **term vector space** W is defined as:

if term k does not occur in document d_i, $w_{ik} = 0$

if term k occurs in document d_i, w_{ik} is greater than zero

(w_{ik} is called the **weight** of term k in document d_i)

Similarity between d_i and d_j is defined as:

$$\cos(\mathbf{d}_i, \mathbf{d}_j) = \frac{\sum_{k=1}^{n} w_{ik} w_{jk}}{|\mathbf{d}_i|\ |\mathbf{d}_j|}$$

Where $\mathbf{d}_i$ and $\mathbf{d}_j$ are the corresponding weighted term vectors

Simple Uses of Vector Similarity in Information Retrieval

Threshold

For query *q*, retrieve all documents with similarity above a threshold, e.g., similarity > 0.50.

Ranking

For query *q*, return the *n* most similar documents ranked in order of similarity.

[This is the standard practice.]

Simple Example of Ranking (Weighting by Term Frequency)

query		
q	*ant dog*	
document	**text**	**terms**
d_1	*ant ant bee*	*ant bee*
d_2	*dog bee dog hog dog ant dog*	*ant bee dog hog*
d_3	*cat gnu dog eel fox*	*cat dog eel fox gnu*

	ant								*length*
q									$\sqrt{2}$
d_1									$\sqrt{5}$
d_2	1	1						1	$\sqrt{19}$
d_3			1	1	1	1	1		$\sqrt{5}$

Now compute the similarities of these documents with *q*?

Calculate Ranking

Similarity of query to documents in example:

	d_1	d_2	d_3
q	$2/\sqrt{10}$ 0.63	$5/\sqrt{38}$ 0.81	$1/\sqrt{10}$ 0.32

If the query q is searched against this document set, the ranked results are:

d_2, d_1, d_3

Cosine similarity amongst 3 documents

How similar are the novels

SaS: *Sense and Sensibility*

PaP: *Pride and Prejudice*, and

WH: *Wuthering Heights*?

term	SaS	PaP	WH
affection	115	58	20
jealous	10	7	11
gossip	2	0	6
wuthering	0	0	38

Term frequencies (counts)

3 documents example contd.

Log frequency weighting

term	SaS	PaP	WH
affection	3.06	2.76	2.30
jealous	2.00	1.85	2.04
gossip	1.30	0	1.78
wuthering	0	0	2.58

After length normalization

term	SaS	PaP	WH
affection	0.789	0.832	0.524
jealous	0.515	0.555	0.465
gossip	0.335	0	0.405
wuthering	0	0	0.588

cos(SaS,PaP) ≈
$0.789 \times 0.832 + 0.515 \times 0.555 + 0.335 \times 0.0 + 0.0 \times 0.0$
≈ 0.94

cos(SaS,WH) ≈ 0.79

cos(PaP,WH) ≈ 0.69

Vector Space Revision

$\mathbf{x} = (x_1, x_2, x_3, ..., x_n)$ is a vector in an n-dimensional vector space

Length of $\mathbf{x}$ is given by (extension of Pythagoras's theorem)

$$|\mathbf{x}|^2 = x_1{}^2 + x_2{}^2 + x_3{}^2 + ... + x_n{}^2$$

If $\mathbf{x}_1$ and $\mathbf{x}_2$ are vectors:

Inner product (or dot product) is given by

$$\mathbf{x}_1 \bullet \mathbf{x}_2 = x_{11}x_{21} + x_{12}x_{22} + x_{13}x_{23} + ... + x_{1n}x_{2n}$$

Cosine of the angle between the vectors $\mathbf{x}_1$ and $\mathbf{x}_{2:}$

$$\cos(\theta) = \frac{\mathbf{x}_1 \bullet \mathbf{x}_2}{|\mathbf{x}_1|\ |\mathbf{x}_2|}$$

Standard vector space, summary

- Very simple
  - Map everything to a vector
  - Compare using angle between vectors
- Challenge is mostly finding good weighting scheme
  - Variants on tf·idf are most common
  - Model provides no guidance
- Another challenge is comparison function
  - Cosine comparison is most common
  - Generic inner product (without unit vectors) also occurs
  - Too many dimensions for storage and computing
  - Redundant basis vectors caused by non-independent terms

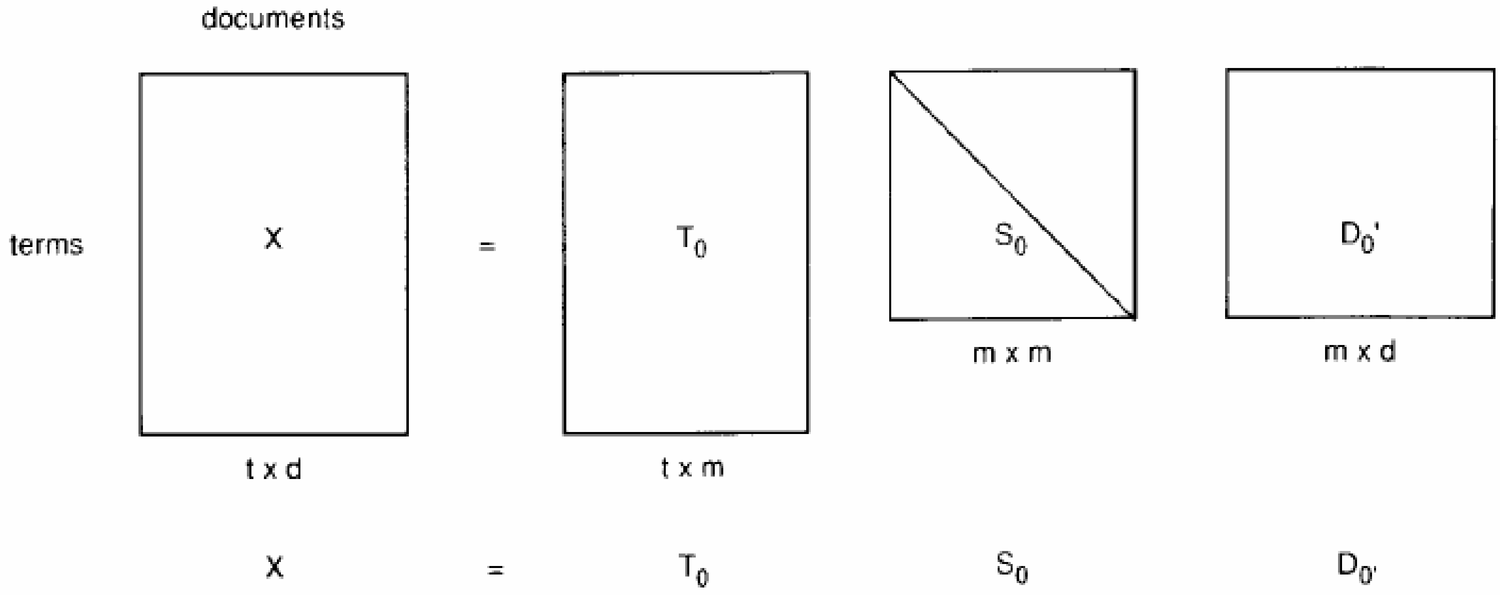
Models we' ll consider

- *Boolean (exact match)*
- *Statistical language models*
- *Vector space*
- Latent Semantic Indexing

Latent Semantic Indexing (LSI)

- One variant of the vector space model
- Use Singular Value Decomposition to identify uncorrelated, significant basis vectors or factors
  - Rather than non-independent terms
- Replace original words with a subset of the new factors (say 100) in both documents and queries
- Compute similarities in this new space
- Computationally expensive, uncertain effectiveness

LSI



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 ($\leq \min(t,d)$)

LSI: example

Technical Memo Example

Titles

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

Terms	Documents								
	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

LSI: example (2)

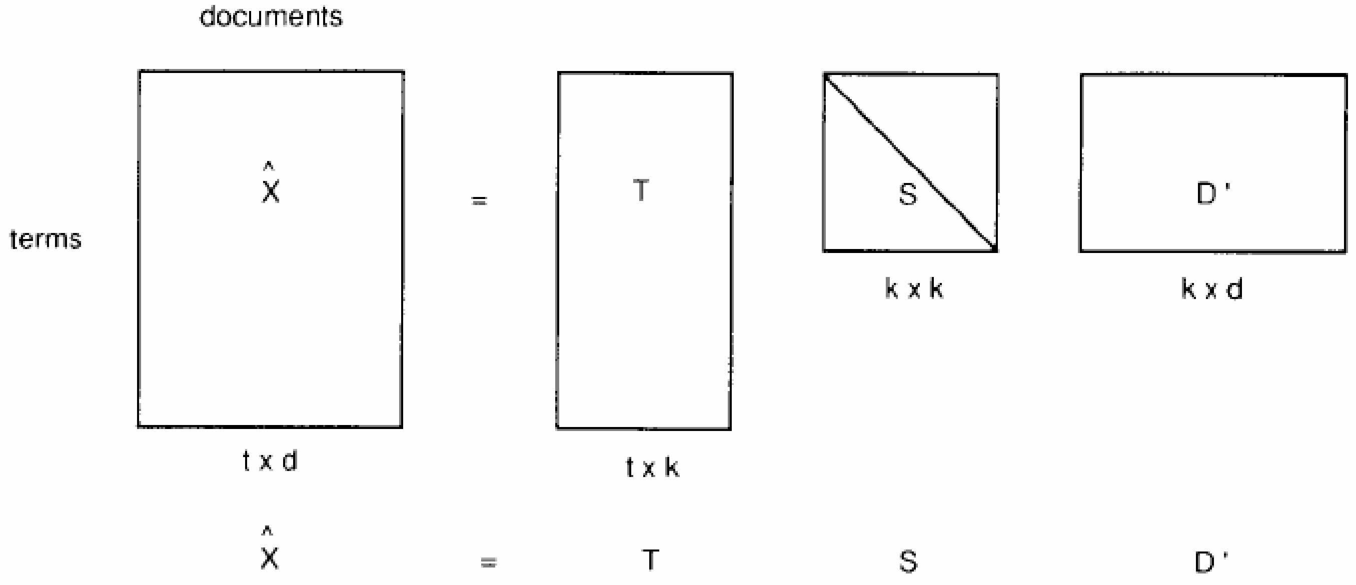
$$T_0 = \begin{bmatrix}
0.22 & -0.11 & 0.29 & -0.41 & -0.11 & -0.34 & 0.52 & -0.06 & -0.41 \\
0.20 & -0.07 & 0.14 & -0.55 & 0.28 & 0.50 & -0.07 & -0.01 & -0.11 \\
0.24 & 0.04 & -0.16 & -0.59 & -0.11 & -0.25 & -0.30 & 0.06 & 0.49 \\
0.40 & 0.06 & -0.34 & 0.10 & 0.33 & 0.38 & 0.00 & 0.00 & 0.01 \\
0.64 & -0.17 & 0.36 & 0.33 & -0.16 & -0.21 & -0.17 & 0.03 & 0.27 \\
0.27 & 0.11 & -0.43 & 0.07 & 0.08 & -0.17 & 0.28 & -0.02 & -0.05 \\
0.27 & 0.11 & -0.43 & 0.07 & 0.08 & -0.17 & 0.28 & -0.02 & -0.05 \\
0.30 & -0.14 & 0.33 & 0.19 & 0.11 & 0.27 & 0.03 & -0.02 & -0.17 \\
0.21 & 0.27 & -0.18 & -0.03 & -0.54 & 0.08 & -0.47 & -0.04 & -0.58 \\
0.01 & 0.49 & 0.23 & 0.03 & 0.59 & -0.39 & -0.29 & 0.25 & -0.23 \\
0.04 & 0.62 & 0.22 & 0.00 & -0.07 & 0.11 & 0.16 & -0.68 & 0.23 \\
0.03 & 0.45 & 0.14 & -0.01 & -0.30 & 0.28 & 0.34 & 0.68 & 0.18
\end{bmatrix}$$

$$S_0 = \begin{bmatrix}
3.34 & & & & & & & & \\
 & 2.54 & & & & & & & \\
 & & 2.35 & & & & & & \\
 & & & 1.64 & & & & & \\
 & & & & 1.50 & & & & \\
 & & & & & 1.31 & & & \\
 & & & & & & 0.85 & & \\
 & & & & & & & 0.36 & \\
 & & & & & & & & 0.36
\end{bmatrix}$$

Decreasing order

$$D_0 = \begin{bmatrix}
0.20 & -0.06 & 0.11 & -0.95 & 0.05 & -0.08 & 0.18 & -0.01 & -0.06 \\
0.61 & 0.17 & -0.50 & -0.03 & -0.21 & -0.26 & -0.43 & 0.05 & 0.24 \\
0.46 & -0.03 & 0.21 & 0.04 & 0.38 & 0.72 & -0.24 & 0.01 & 0.02 \\
0.54 & -0.23 & 0.57 & 0.27 & -0.21 & -0.37 & 0.26 & -0.02 & -0.08 \\
0.28 & 0.11 & -0.51 & 0.15 & 0.33 & 0.03 & 0.67 & -0.06 & -0.26 \\
0.00 & 0.19 & 0.10 & 0.02 & 0.39 & -0.30 & -0.34 & 0.45 & -0.62 \\
0.01 & 0.44 & 0.19 & 0.02 & 0.35 & -0.21 & -0.15 & -0.76 & 0.02 \\
0.02 & 0.62 & 0.25 & 0.01 & 0.15 & 0.00 & 0.25 & 0.45 & 0.52 \\
0.08 & 0.53 & 0.08 & -0.03 & -0.60 & 0.36 & -0.04 & -0.07 & -0.45
\end{bmatrix}$$

LSI



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

T has orthogonal, unit-length columns (T' T = I)
D has orthogonal, unit-length columns (D' 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)

LSI: example for ” k=2”

$$X \approx \underset{T}{\begin{bmatrix} 0.22 & -0.11 \\ 0.20 & -0.07 \\ 0.24 & 0.04 \\ 0.40 & 0.06 \\ 0.64 & -0.17 \\ 0.27 & 0.11 \\ 0.27 & 0.11 \\ 0.30 & -0.14 \\ 0.21 & 0.27 \\ 0.01 & 0.49 \\ 0.04 & 0.62 \\ 0.03 & 0.45 \end{bmatrix}} \underset{S}{\begin{bmatrix} 3.34 & \\ & 2.54 \end{bmatrix}} \underset{D'}{\begin{bmatrix} 0.20 & 0.61 & 0.46 & 0.54 & 0.28 & 0.00 & 0.02 & 0.02 & 0.08 \\ -0.06 & 0.17 & -0.13 & -0.23 & 0.11 & 0.19 & 0.44 & 0.62 & 0.53 \end{bmatrix}}$$

LSI: example (3)

$$\hat{X} =$$

$$\begin{matrix}
0.16 & 0.40 & 0.38 & 0.47 & 0.18 & -0.05 & -0.12 & -0.16 & -0.09 \\
0.14 & 0.37 & 0.33 & 0.40 & 0.16 & -0.03 & -0.07 & -0.10 & -0.04 \\
0.15 & 0.51 & 0.36 & 0.41 & 0.24 & 0.02 & 0.06 & 0.09 & 0.12 \\
0.26 & 0.84 & 0.61 & 0.70 & 0.39 & 0.03 & 0.08 & 0.12 & 0.19 \\
0.45 & 1.23 & 1.05 & 1.27 & 0.56 & -0.07 & -0.15 & -0.21 & -0.05 \\
0.16 & 0.58 & 0.38 & 0.42 & 0.28 & 0.06 & 0.13 & 0.19 & 0.22 \\
0.16 & 0.58 & 0.38 & 0.42 & 0.28 & 0.06 & 0.13 & 0.19 & 0.22 \\
0.22 & 0.55 & 0.51 & 0.63 & 0.24 & -0.07 & -0.14 & -0.20 & -0.11 \\
0.10 & 0.53 & 0.23 & 0.21 & 0.27 & 0.14 & 0.31 & 0.44 & 0.42 \\
-0.06 & 0.23 & -0.14 & -0.27 & 0.14 & 0.24 & 0.55 & 0.77 & 0.66 \\
-0.06 & 0.34 & -0.15 & -0.30 & 0.20 & 0.31 & 0.69 & 0.98 & 0.85 \\
-0.04 & 0.25 & -0.10 & -0.21 & 0.15 & 0.22 & 0.50 & 0.71 & 0.62
\end{matrix}$$

Comparing original and LSI

	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

human	0.16	0.40	0.38	0.47	0.18	−0.05	−0.12	−0.16	−0.09
interface	0.14	0.37	0.33	0.40	0.16	−0.03	−0.07	−0.10	−0.04
computer	0.15	0.51	0.36	0.41	0.24	0.02	0.06	0.09	0.12
user	0.26	0.84	0.61	0.70	0.39	0.03	0.08	0.12	0.19
system	0.45	1.23	1.05	1.27	0.56	−0.07	−0.15	−0.21	−0.05
response	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
time	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
EPS	0.22	0.55	0.51	0.63	0.24	−0.07	−0.14	−0.20	−0.11
survey	0.10	0.53	0.23	0.21	0.27	0.14	0.31	0.44	0.42
trees	−0.06	0.23	−0.14	−0.27	0.14	0.24	0.55	0.77	0.66
graph	−0.06	0.34	−0.15	−0.30	0.20	0.31	0.69	0.98	0.85
minors	−0.04	0.25	−0.10	−0.21	0.15	0.22	0.50	0.71	0.62

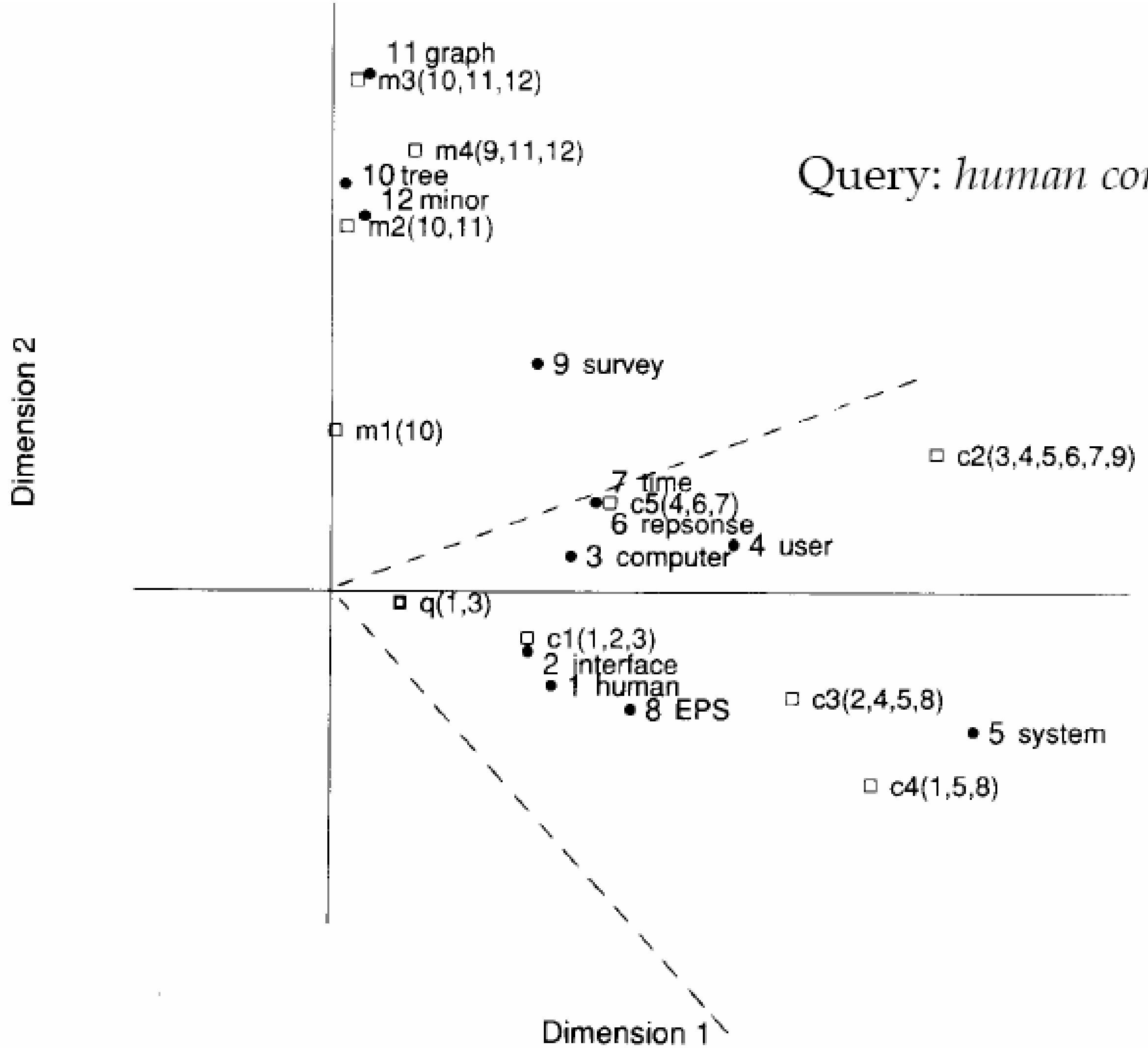
Using LSI

$$X \approx \underbrace{\begin{array}{rr} 0.22 & -0.11 \\ 0.20 & -0.07 \\ 0.24 & 0.04 \\ 0.40 & 0.06 \\ 0.64 & -0.17 \\ 0.27 & 0.11 \\ 0.27 & 0.11 \\ 0.30 & -0.14 \\ 0.21 & 0.27 \\ 0.01 & 0.49 \\ 0.04 & 0.62 \\ 0.03 & 0.45 \end{array}}_{T} \; \underbrace{\begin{array}{rr} 3.34 & \\ & 2.54 \end{array}}_{S} \; \underbrace{\begin{array}{rrrrrrrrr} 0.20 & 0.61 & 0.46 & 0.54 & 0.28 & 0.00 & 0.02 & 0.02 & 0.08 \\ -0.06 & 0.17 & -0.13 & -0.23 & 0.11 & 0.19 & 0.44 & 0.62 & 0.53 \end{array}}_{D'}$$

- D is new doc vectors (2 dimensions, here)
- T provides term vectors
- Given $Q=q_1q_2\ldots q_t$ want to compare to docs
- Convert Q from *t* dimensions to 2
  - $Q' = Q^T\, T\, S^{-1}$
  - $Q'\ (1\times k) = Q^T\ (1\times t)\ T\ (t\times k)\ S^{-1}\ (k\times k)$
- Can now compare to doc vectors

- Same basic approach can be used to add new docs to the database

LSI



Query: *human computer interaction*

Is LSI any good?

- Decomposes language into “basis vectors”
  - In a sense, is looking for core concepts
- In theory, this means that system will retrieve documents using synonyms of your query words
  - The “magic” that appeals to people
- From a demo at http://lsi.research.telcordia.com
  - They hold the patent on LSI
- Query “manna” (以色列人漂泊荒野时上帝所赐的食物) on Bible verses (312 dimensions)
  - #5 – Exodus (出埃及记) 12_20 Ye shall eat nothing leavened; in all your habitations shall ye eat unleavened(未发酵的) bread.
  - #6 -- Genesis 31_54 Then Jacob offered sacrifice upon the mount, and called his brethren to eat bread: and they did eat bread, and tarried all night in the mount.
- Things like this are major claim of LSI techniques

Magic can be confusing

- Top 5 hits for query “apple” (312 dimensions)
  - *Song_of_Songs 8_5* Who is this that cometh up from the wilderness, leaning upon her beloved? I raised thee up under the **apple** tree: there thy mother brought thee forth: there she brought thee forth that bare thee.
  - *Psalms 47_3* He shall subdue the people under us, and the nations under our feet. **????**
  - *Song_of_Songs* 2_3 As the **apple** tree among the trees of the wood, so is my beloved among the sons. I sat down under his shadow with great delight, and his fruit was sweet to my taste.
  - *Zecharaiah 3_10* In that day, saith the LORD of hosts, shall ye call every man his neighbour under the vine and under the fig tree(无花果树). **Magic?**
  - *Ecclesiastes 4_7* Then I returned, and I saw vanity under the sun. **????**

Vector Space Retrieval Model: Summary

- Standard vector space
  - Each dimension corresponds to a term in the vocabulary
  - Vector elements are real-valued, reflecting term importance
  - Any vector (document, query, ...) can be compared to any other
  - Cosine correlation is the similarity metric used most often
- Latent Semantic Indexing (LSI)
  - Each dimension corresponds to a "basic concept"
  - Documents and queries mapped into basic concepts
  - Same as standard vector space after that
  - Whether it's good depends on what you want

Vector Space Model: Disadvantages

- Assumed independence relationship among terms
  - Though this is a *very* common retrieval model assumption
- Lack of justification for some vector operations
  - e.g. choice of similarity function
  - e.g., choice of term weights
- Barely a retrieval model
  - Doesn’t explicitly model relevance, a person’s information need, language models, etc.
- Assumes a query and a document can be treated the same (symmetric)
- Lack of a cognitive (or other) justification

Vector Space Model: Advantages

- Simplicity
- Ability to incorporate term weights
  - *Any* type of term weights can be added
  - No model that has to justify the use of a weight
- Can measure similarities between
  - documents and queries
  - documents and documents
  - queries and queries
  - sentences and sentences
  - etc.

Homework 03

Backup

Efficiency of VSM
A glance for implementation

- Problems: how to implement efficient cosine ranking?
- Basic Assumption:
  - Find out *K* best documents that match a query rather than rank all documents

Efficient cosine ranking

- Find the K docs in the collection "nearest" to the query $\Rightarrow$ K largest query-doc cosines.
- Efficient ranking:
  - Computing a single cosine efficiently.
  - Choosing the K largest cosine values efficiently.
    - Can we do this without computing all N cosines?

Efficient cosine ranking

- What we're doing in effect: solving the *K*-nearest neighbor problem for a query vector
- In general, we do not know how to do this efficiently for high-dimensional spaces
- But it is solvable for short queries, and standard indexes support this well, and we just take a glance for some of the solutions since Indexing is still not discussed.

Special case – unweighted queries

- No weighting on query terms
  - Assume each query term occurs only once
- Then for ranking, don't need to normalize query vector
  - Slight simplification of algorithm

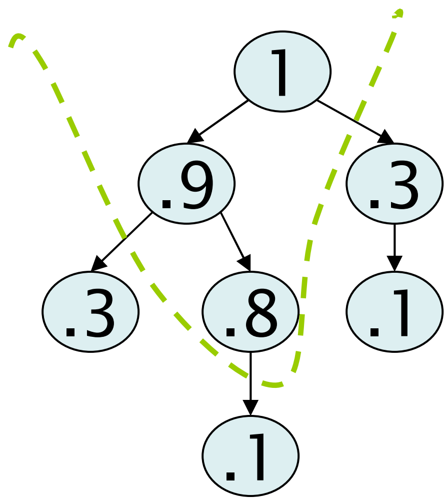
Computing the *K* largest cosines: selection vs. sorting

- Typically we want to retrieve the top *K* docs (in the cosine ranking for the query)
  - not to totally order all docs in the collection
- Can we pick off docs with *K* highest cosines?
- Let *J* = number of docs with nonzero cosines
  - We seek the *K* best of these *J*

Use heap for selecting top *K*

- Binary tree in which each node's value > the values of children
- Takes *2J* operations to construct (the time complexity for optimized construction of binary heap*), then each of *K* "winners" read off in 2log *J* steps.
- For *J*=1M, *K*=100, this is about 10% of the cost of sorting.

* http://en.wikipedia.org/wiki/Binary_heap



Bottlenecks

- Primary computational bottleneck in scoring: <u>cosine computation</u>
- Can we avoid all this computation?
- Yes, but may sometimes get it wrong
  - a doc *not* in the top *K* may creep into the list of *K* output docs
  - Is this such a bad thing?

Cosine similarity is only a proxy

- User has a task and a query formulation
- Cosine matches docs to query
- Thus cosine is anyway a proxy for user happiness
- If we get a list of *K* docs "close" to the top *K* by cosine measure, should be ok

Generic approach

- Find a set A of *contenders*, with $K < |A| << N$
  - A does not necessarily contain the top K, but has many docs from among the top K
  - Return the top K docs in A
- Think of A as pruning non-contenders
- The same approach is also used for other (non-cosine) scoring functions
- Will look at several schemes following this approach

Index elimination

- Basic algorithm
  - cosine computation algorithm only considers docs containing at least one query term
- Take this further:
  - Only consider high-idf query terms
  - Only consider docs containing many query terms

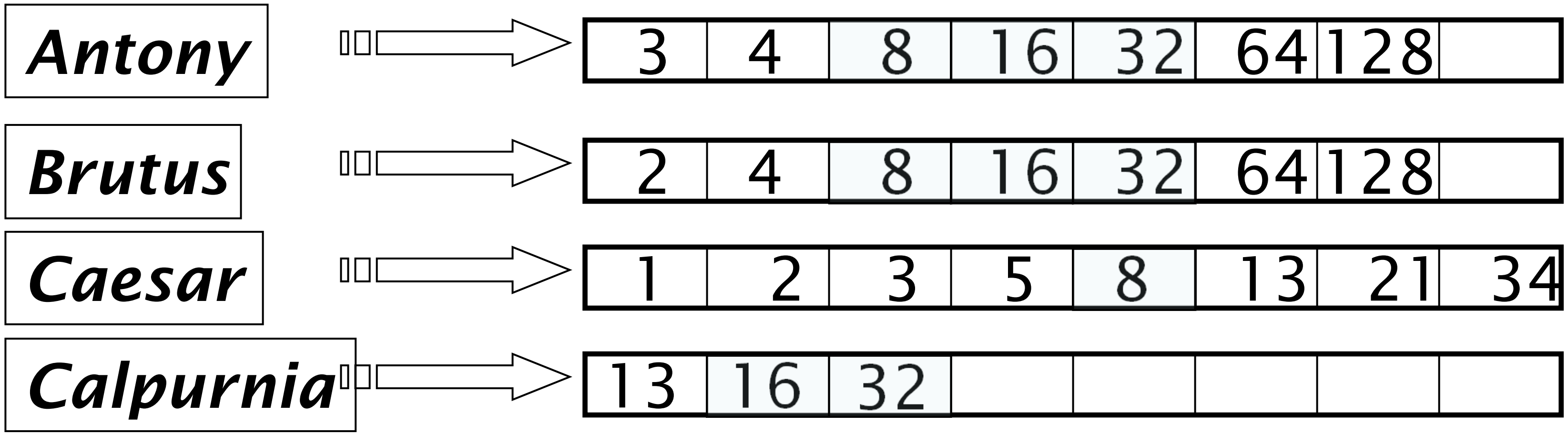
High-idf query terms only

- For a query such as *"catcher in the rye"*
- Only accumulate scores from *catcher* and *rye*
- Intuition: ***in*** and ***the*** contribute little to the scores and so don't alter rank-ordering much
- Benefit:
  – Postings of low-idf terms have many docs → these (many) docs get eliminated from set *A* of contenders

Docs containing many query terms

- Any doc with at least one query term is a candidate for the top *K* output list
- For multi-term queries, only compute scores for docs containing several of the query terms
  - Say, at least 3 out of 4
  - Imposes a "soft conjunction" on queries seen on web search engines (early Google)
- Easy to implement in postings traversal

3 of 4 query terms



Scores only computed for docs 8, 16 and 32.