Text Processing on the Web

Week 4

Dimensionality Reduction:
LSI and pLSI

The material for these slides are borrowed heavily from the precursor of this course by Tat-Seng Chua as well as slides from the accompanying recommended texts Baldi et al. and Manning et al.

Recap

- Probabilistic Model
  - +: Based on a firm theoretical foundation; justified optimal ranking
  - -: Binary word-in-doc weights (not using term frequencies)
    Independence of terms (can be alleviated)
    Has never worked convincingly better in practice

- Language Model
  - Accounts for term frequency and document length within model
  - But based in probability so accounting is different
  - n-gram models possible, but unigram easy and still useful
  - Like VSM, puts queries and documents as same types of objects

Outline

- Synonymy and Polysemy
- Bit of Linear Algebra
- Latent Semantic Indexing
- pLSI

Problems with Lexical Semantics

- Ambiguity and association in natural language
  - **Polysemy**: Words often have a **multitude of meanings** and different types of usage *(more severe in very heterogeneous collections).*
  - The vector space model is unable to discriminate between different meanings of the same word.

$$\mathrm{sim}_{\mathrm{true}}(d, q) < \cos(\angle(\vec{d}, \vec{q}))$$

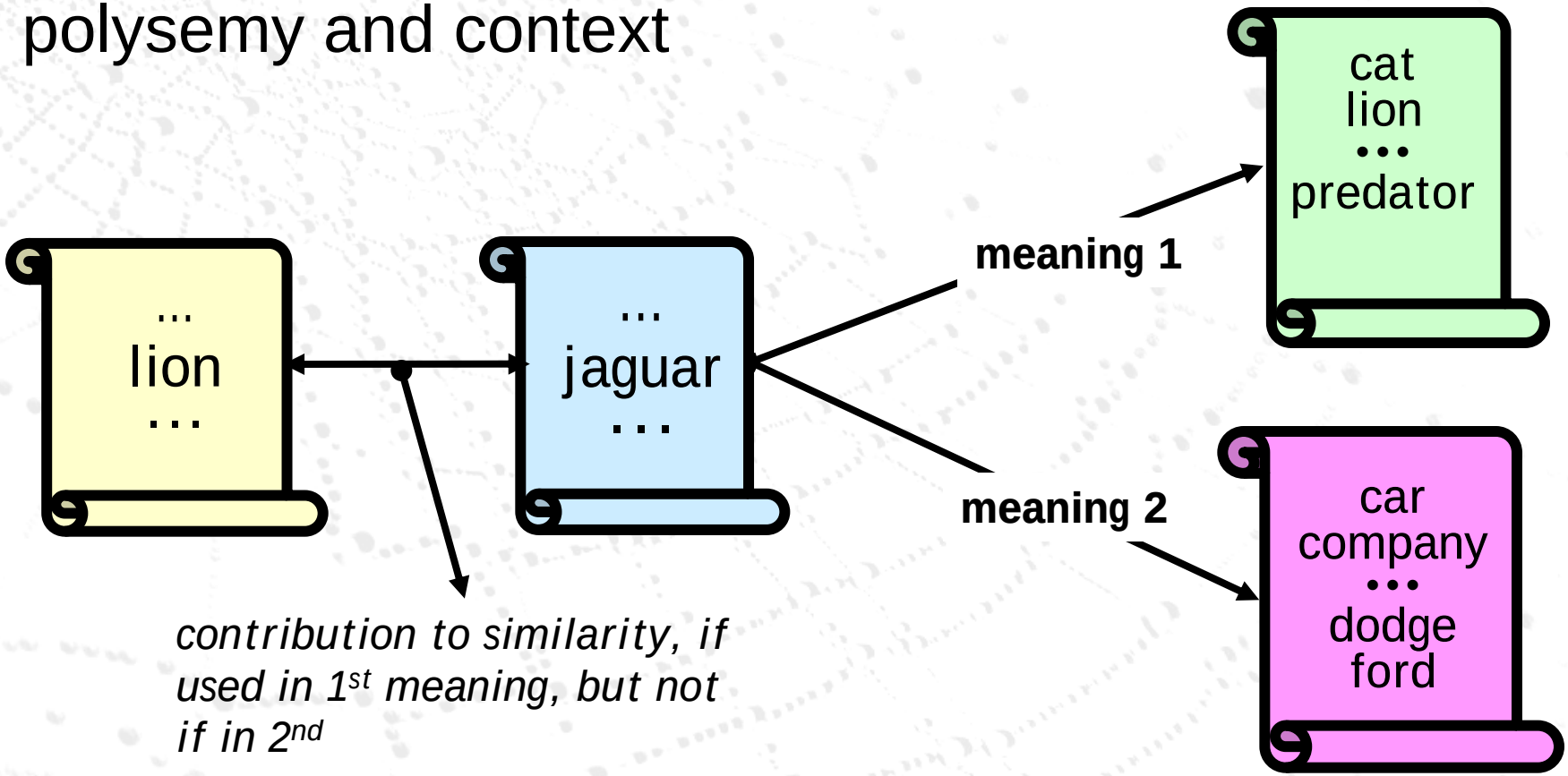
Problems with Lexical Semantics

- **Synonymy**: Different terms may have an **identical or a similar meaning** (weaker: words indicating the same topic).
- No associations between words are made in the vector space representation.

$$\mathrm{sim}_{\mathrm{true}}(d,q) > \cos(\angle(\vec{d},\vec{q}))$$

Polysemy and Context

- Document similarity on single word level: polysemy and context



Singular Value Decomposition

For an $m \times n$ matrix $\mathbf{A}$ of rank r there exists a factorization (Singular Value Decomposition = **SVD**) as follows:

$$A = U \Sigma V^T$$

$m \times m$ $\quad$ $m \times n$ $\quad$ V is $n \times n$

The columns of $\boldsymbol{U}$ are orthogonal eigenvectors of $\boldsymbol{AA^T}$.

The columns of $\boldsymbol{V}$ are orthogonal eigenvectors of $\boldsymbol{A^TA}$.

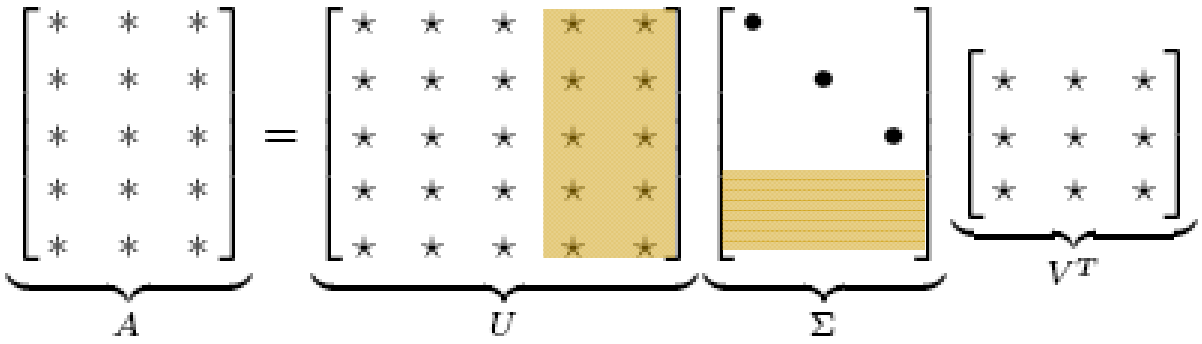
Eigenvalues $\lambda_1 \ldots \lambda_r$ of $\boldsymbol{AA^T}$ are the eigenvalues of $\boldsymbol{A^TA}$.

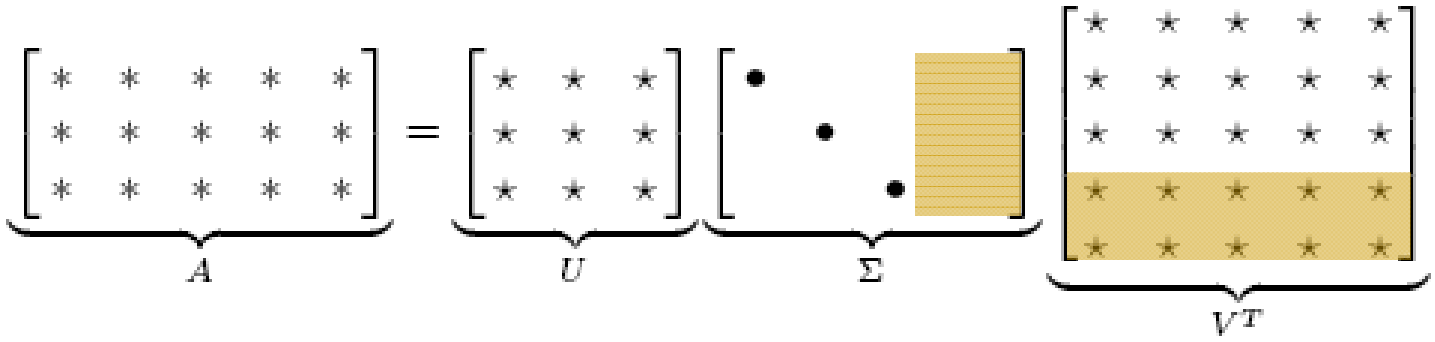
$$\sigma_i = \sqrt{\lambda_i}$$

$$\Sigma = diag(\sigma_1 \ldots \sigma_r)$$ ← *Singular values.*

Singular Value Decomposition

- Illustration of SVD dimensions and sparseness





SVD example

Let $A = \begin{bmatrix} 1 & -1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$

Thus m=3, n=2. Its SVD is

$$\begin{bmatrix} 0 & 2/\sqrt{6} & 1/\sqrt{3} \\ 1/\sqrt{2} & -1/\sqrt{6} & 1/\sqrt{3} \\ 1/\sqrt{2} & 1/\sqrt{6} & -1/\sqrt{3} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{3} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}$$

Typically, the singular values arranged in decreasing order.

Low-rank Approximation

- SVD can be used to compute optimal **low-rank approximations**.
- Approximation problem: Find $\boldsymbol{A_k}$ of rank $\boldsymbol{k}$ such that $A_k = \min_{X:rank(X)=k} \|A - X\|_F$ ← *Frobenius norm*

$$\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} |a_{ij}|^2}.$$

- A_k and X are both $m \times n$ matrices.

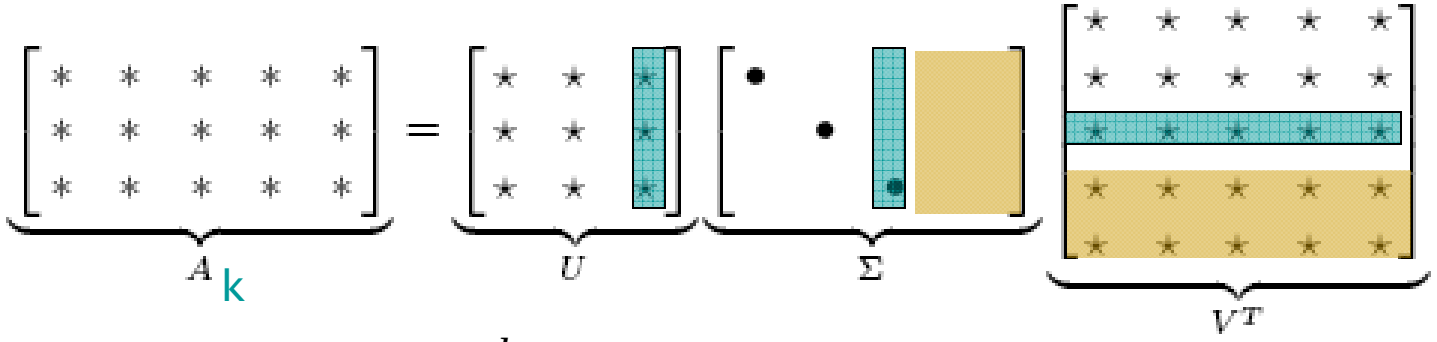
Typically, want $k << r$.

Low-rank Approximation

- Solution via SVD

set smallest r-k singular values to zero

$$A_k = U\ \mathrm{diag}(\sigma_1,...,\sigma_k,\underbrace{0,...,0}) V^T$$



$$A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T$$

column notation: sum of rank 1 matrices

Approximation error

- How good (bad) is this approximation?
- It's the best possible, measured by the Frobenius norm of the error:

$$\min_{X:rank(X)=k} \|A-X\|_F = \|A-A_k\|_F = \sigma_{k+1}$$

where the σ_i are ordered such that $\sigma_i \geq \sigma_{i+1}$.

Suggests why Frobenius error drops as *k* is increased.

SVD Low-rank approximation

- Whereas the term-doc matrix *A* may have *m*=50000, *n*=10 million (and rank close to 50000)
- We can construct an approximation A_{100} with rank 100.
  - Of all rank 100 matrices, it would have the lowest Frobenius error.
- Great … but why would we??
- Answer: *Latent Semantic Indexing*

Latent Semantic Analysis (LSA)

- LSA aims to discover something about the meaning behind the words; about the topics in the documents.
- What is the difference between topics and words?
  - Words are observable
  - Topics are not. They are latent.
- How to find out topics from the words in an automatic way?
  - We can imagine them as a combination of words

Goals of LSI

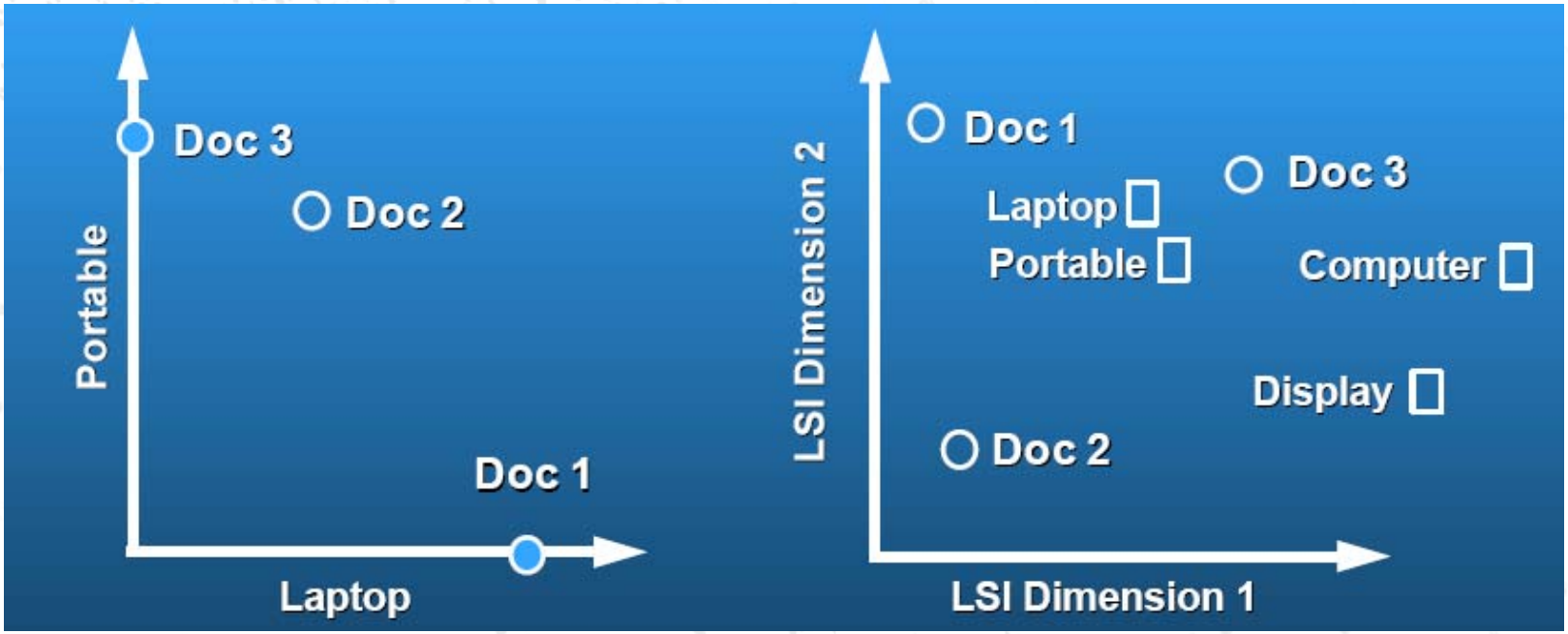
- Similar terms map to similar location in low dimensional space
- Noise reduction by dimension reduction

Latent Semantic Indexing (LSI)

- Perform a **low-rank approximation** of **document-term matrix** (typical rank **100-300**)
- General idea
  - Map documents (*and* terms) to a **low-dimensional** representation.
  - Design a mapping such that the low-dimensional space reflects **semantic associations** (latent semantic space).
  - Compute document similarity based on the **inner product** in this **latent semantic space**

Latent Semantic Analysis

- **Latent semantic space**: illustrating example



courtesy of Susan Dumais

Performing the maps

- Each row and column of *A* gets mapped into the *k*-dimensional LSI space, by the SVD.
- Claim – this is not only the mapping with the best (Frobenius error) approximation to *A*, but in fact *improves* retrieval.
- A query q is also mapped into this space, by

$$q_k = q^T U_k \Sigma_k^{-1}$$

  - Query NOT a sparse vector.

LSI Example

m=5 (interface, library, Java, Kona, blend), n=7

$$A = \begin{pmatrix} 1&2&1&5&0&0&0 \\ 1&2&1&5&0&0&0 \\ 1&2&1&5&0&0&0 \\ 0&0&0&0&2&3&1 \\ 0&0&0&0&2&3&1 \end{pmatrix} = \begin{pmatrix} 0.58 & 0.00 \\ 0.58 & 0.00 \\ 0.58 & 0.00 \\ 0.00 & 0.71 \\ 0.00 & 0.71 \end{pmatrix} \times \begin{pmatrix} 9.64 & 0.00 \\ 0.00 & 5.29 \end{pmatrix} \times \begin{pmatrix} 0.18 & 0.36 & 0.18 & 0.90 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.53 & 0.80 & 0.27 \end{pmatrix}$$

U $\quad$ Δ $\quad$ V^T

- a query $q = (0\ 0\ 1\ 0\ 0)^T$ is transformed into $q' = U^T \times q = (0.58\ \ 0.00)^T$ and evaluated on V^T
- a new document $d_8 = (1\ 1\ 0\ 0\ 0)^T$ is transformed into $d_8' = U^T \times d_8 = (1.16\ \ 0.00)^T$ and appended to V^T

Probabilistic LSA

Probabilistic LSA

- Let us start from what we know
- Remember the BoW model

$$P(doc) = P(term_1 \mid doc)P(term_2 \mid doc)...P(term_L \mid doc)$$

$$= \prod_{l=1}^{L} P(term_l \mid doc) = \prod_{t=1}^{T} P(term_t \mid doc)^{X(term_t, doc)}$$

We know how to compute the parameter of this model, i.e. $P(term_t|doc)$

From our last Prob IR and LM IR lecture, remember?

One way: Maximum Likelihood

Probabilistic LSA

Unobserved, latent variable

- Now let us have *k* topics as well:

$$P(term_t \mid doc) = \sum_{k=1}^{K} P(term_t \mid topic_k) P(topic_k \mid doc)$$

The same (written using shorthand)

$$P(t \mid doc) = \sum_{k=1}^{K} P(t \mid k) P(k \mid doc)$$

So by replacing this, for any doc in the collection,

$$P(doc) = \prod_{t=1}^{T} \{ \sum_{k=1}^{K} P(t \mid k) P(k \mid doc) \}^{X(t,doc)}$$

Which are the parameters of this model?

Probabilistic LSA

- The parameters of this model are:
  - P(t|k)
  - P(k|doc)
- It is possible to derive the equations for computing these parameters by Maximum Likelihood (again)
- If we do so, what do we get?
  - P(t|k) for all t and k, is a term by topic matrix
    (gives which terms make up a topic)
  - P(k|doc) for all k and doc, is a topic by document matrix
    (gives which topics are in a document)

Deriving the parameter estimation algorithm

- The log likelihood of this model is the log probability of the entire collection:

$$\sum_{d=1}^{N} \log P(d) = \sum_{d=1}^{N} \sum_{t=1}^{T} X(t,d) \log \sum_{k=1}^{K} P(t \mid k) P(k \mid d)$$

which is to be maximised w.r.t. parameters $P(t \mid k)$ and then also $P(k \mid d)$,

subject to the constraints that $\sum_{t=1}^{T} P(t \mid k) = 1$ and $\sum_{k=1}^{K} P(k \mid d) = 1.$

The pLSA algorithm

- Inputs: term by document matrix X(t,d), t=1…T, d=1…N and the number K of topics sought
- **Initialise** arrays P1 and P2 randomly (between [0,1]) and normalise them to sum to 1 along rows
- **Iterate** until convergence
  - For d=1 to N, for t =1 to T, for k=1:K

Variant of EM!

$$P1(t,k) \leftarrow P1(t,k)\sum_{d=1}^{N}\frac{X(t,d)}{\sum_{k=1}^{K}P1(t,k)P2(k,d)}P2(k,d); \quad P1(t,k) \leftarrow \frac{P1(t,k)}{\sum_{t=1}^{T}P1(t,k)}$$

$$P2(k,d) \leftarrow P2(k,d)\sum_{t=1}^{T}\frac{x(t,d)}{\sum_{k=1}^{K}P1(t,k)P2(k,d)}P1(t,k); \quad P2(k,d) \leftarrow \frac{P2(k,d)}{\sum_{k=1}^{K}P2(k,d)}$$

- Output: arrays P1 and P2, which hold the estimated parameters P(t|k) and P(k|d) respectively

Experimental Results

Empirical evidence

- Experiments on Text REtrieval Conference data
  - Running times of ~ one day on tens of thousands of docs
- Dimensions – various values 250-350 reported
  - (Under 200 reported unsatisfactory)
- Generally expect recall to improve – what about precision?

Empirical evidence

- Precision at or above median TREC precision
  - Top scorer on almost 20% of TREC topics
- Slightly better on average than straight vector spaces
- Effect of dimensionality:

Dimensions	Precision
250	0.367
300	0.371
346	0.374

Some wild extrapolation

- The “dimensionality” of a corpus is the number of distinct topics represented in it.

- More mathematically “wild” extrapolation:
  - if *A* has a rank *k* approximation of low Frobenius error, then there are ~~exactly~~ no more than *k* distinct topics in the corpus.

LSI and other applications

- In many settings in pattern recognition and retrieval, we have a feature-object matrix.
  - For text, the terms are features and the docs are objects.
  - Could be opinions and users …
  - This matrix may be redundant in dimensionality.
  - Can work with low-rank approximation.
  - If entries are missing (e.g., users' opinions), can recover if dimensionality is low.

- Powerful general analytical technique
  - Close, principled analog to **clustering** methods.

Return to this in a later lecture

"Arts"	"Budgets"	"Children"	"Education"
NEW	MILLION	CHILDREN	SCHOOL
FILM	TAX	WOMEN	STUDENTS
SHOW	PROGRAM	PEOPLE	SCHOOLS
MUSIC	BUDGET	CHILD	EDUCATION
MOVIE	BILLION	YEARS	TEACHERS
PLAY	FEDERAL	FAMILIES	HIGH
MUSICAL	YEAR	WORK	PUBLIC
BEST	SPENDING	PARENTS	TEACHER
ACTOR	NEW	SAYS	BENNETT
FIRST	STATE	FAMILY	MANIGAT
YORK	PLAN	WELFARE	NAMPHY
OPERA	MONEY	MEN	STATE
THEATER	PROGRAMS	PERCENT	PRESIDENT
ACTRESS	GOVERNMENT	CARE	ELEMENTARY
LOVE	CONGRESS	LIFE	HAITI

The William Randolph Hearst Foundation will give $1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. "Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services," Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center's share will be $200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive $400,000 each. The Juilliard School, where music and the performing arts are taught, will get $250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual $100,000 donation, too.

Example of topics found from a Science Magazine papers collection

universe	0.0439
galaxies	0.0375
clusters	0.0279
matter	0.0233
galaxy	0.0232
cluster	0.0214
cosmic	0.0137
dark	0.0131
light	0.0109
density	0.01

drug	0.0672
patients	0.0493
drugs	0.0444
clinical	0.0346
treatment	0.028
trials	0.0277
therapy	0.0213
trial	0.0164
disease	0.0157
medical	0.00997

cells	0.0675
stem	0.0478
human	0.0421
cell	0.0309
gene	0.025
tissue	0.0185
cloning	0.0169
transfer	0.0155
blood	0.0113
embryos	0.0111

sequence	0.0818
sequences	0.0493
genome	0.033
dna	0.0257
sequencing	0.0172
map	0.0123
genes	0.0122
chromosome	0.0119
regions	0.0119
human	0.0111

years	0.156
million	0.0556
ago	0.045
time	0.0317
age	0.0243
year	0.024
record	0.0238
early	0.0233
billion	0.0177
history	0.0148

bacteria	0.0983
bacterial	0.0561
resistance	0.0431
coli	0.0381
strains	0.025
microbiol	0.0214
microbial	0.0196
strain	0.0165
salmonella	0.0163
resistant	0.0145

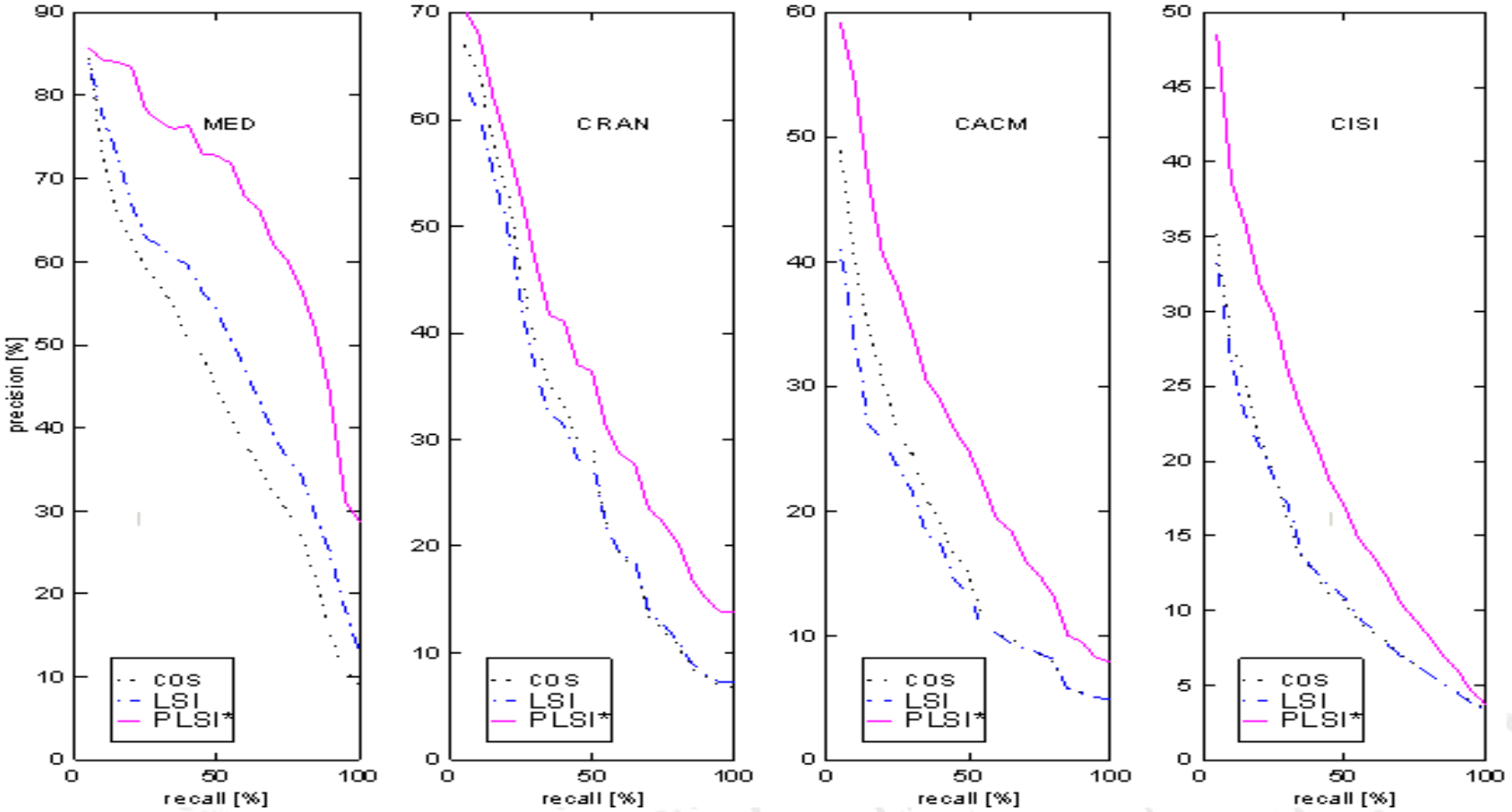
male	0.0558
females	0.0541
female	0.0529
males	0.0477
sex	0.0339
reproductive	0.0172
offspring	0.0168
sexual	0.0166
reproduction	0.0143
eggs	0.0138

theory	0.0811
physics	0.0782
physicists	0.0146
einstein	0.0142
university	0.013
gravity	0.013
black	0.0127
theories	0.01
aps	0.00987
matter	0.00954

immune	0.0909
response	0.0375
system	0.0358
responses	0.0322
antigen	0.0263
antigens	0.0184
immunity	0.0176
immunology	0.0145
antibody	0.014
autoimmune	0.0128

stars	0.0524
star	0.0458
astrophys	0.0237
mass	0.021
disk	0.0173
black	0.0161
gas	0.0149
stellar	0.0127
astron	0.0125
hole	0.00824

The performance of a retrieval system based on this model (PLSI) was found superior to that of both the vector space based similarity (cos) and a non-probabilistic latent semantic indexing (LSI) method.



Cons of LSA-like models

- Negated phrases
  - TREC topics sometimes negate certain query/terms phrases Boolean queries
  - As usual, freetext/vector space syntax of LSI queries precludes (say) “Find any doc having to do with the following 5 companies”
- See the Deerwester *et al.* paper for more.

Summary

- Synonymy and Polysemy affect all standard IR models – not just limited to VSM
- We want to instead model latent (unobserved) topics
  - SVD factors the term-document matrix into orthogonal eigenvectors ("topics"), automatically ranked by salience ("eigenvalue magnitude").
  - LSA does SVD and then drops low order topics to create approximation
  - pLSA does this by taking the unigram LM and injecting a latent variable, k (for k topics)
  - Use maximum likelihood estimation to get probabilities
- Can model fit of approximation using
  - Closely related to clustering, why?

Related resources

- Lost on Linear Algebra wrt SVD? Try: http://www.uwlax.edu/faculty/will/svd/ (great stuff!)
- The BOW toolkit for creating term by doc matrices and other text processing and analysis utilities: http://www.cs.cmu.edu/~mccallum/bow
- SVD is implemented in the SVDPACK software library http://www.netlib.org/svdpack

- Latent Dirichlet Allocation LDA – more powerful version of pLSA
  - Uses a Dirichlet **prior** instead of making a uniform assumption
  - Hence, replace ML with MA**P** for inference