Latent Semantic Indexing

Introduction

- Vector model => use of theory of linear algebra
- Look at matrix formulation
  - M - number of terms in lexicon
  - N - number of documents in collection
  - C the M×N (term×doc.) matrix of weights ≥ 0 *(our old w_{ij})*

$$\begin{pmatrix} c_{11} & \dots & c_{M1} \\ c_{1N} & \dots & c_{MN} \end{pmatrix} \bullet \begin{pmatrix} w_{1q} \\ w_{Mq} \end{pmatrix} = \begin{pmatrix} s_{1q} \\ s_{Nq} \end{pmatrix}$$

document vector — query vector — scores

$s_{xq} = \Sigma^{t}_{i=1}(c_{ix} * w_{iq})$

Introduction

- Vector model => use of theory of linear algebra
- Look at matrix formulation
  - M - number of terms in lexicon
  - N - number of documents in collection
  - C the M×N (term×doc.) matrix of weights ≥ 0 *(our old w_{ij})*

$$\underbrace{\begin{pmatrix} c_{11} & \dots & c_{M1} \\ c_{1N} & \dots & c_{MN} \end{pmatrix}}_{C^T} \bullet \underbrace{\begin{pmatrix} w_{1q} \\ w_{Mq} \end{pmatrix}}_{q} = \begin{pmatrix} s_{1q} \\ s_{Nq} \end{pmatrix}$$

document vector — query vector — scores

$s_{xq} = \Sigma^{t}_{i=1}(c_{ix} * w_{iq})$

Goals

- # terms M large - large dimension
  - ⇒reduce dimension
- find some semantic relationship
  - correlate terms to find structure
    - synonomy
    - polysomy

"people choose same main terms <20% time"

Set-up

C the M×N (term×doc.) matrix of non-negative weights
- of rank r (r ≤ min(M,N))
- documents are *columns of C*

consider CC^T and C^TC:
- symmetric,
- share the same eigenvalues $\lambda_1, \lambda_2, \dots$
  - $\lambda_1, \lambda_2, \dots$ are indexed in decreasing order

- $C^TC(i,j)$ measures similarity documents i and j
- $CC^T(i,j)$ measures strength co-occurrence terms i and j

Use Singular Value Decomposition (SVD)

Theorem:

M×N matrix C of rank r has a *singular value decomposition* $C = U\Sigma V^T$

Where:

U M×M matrix
with columns = orthogonal eigenvectors of CC^T

V N×N matrix
with columns = orthogonal eigenvectors of C^TC

Σ M×N diagonal matrix:
$\Sigma(i,i) = \sqrt{\lambda_i}$ for $1 \le i \le r$
$\Sigma(i,j) = 0$ otherwise

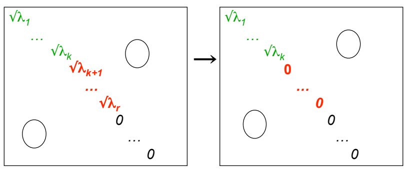
$\sqrt{\lambda_i}$ *called singular values*

$$\begin{pmatrix} \sqrt{\lambda_1} & & & & & \\ & \sqrt{\lambda_2} & & & 0 & \\ & & \dots & & & \\ & & & \sqrt{\lambda_r} & & \\ & 0 & & & 0 & \\ & & & & \dots & \\ & & & & & 0 \end{pmatrix}$$

Reduce Rank

- Reduce rank of Σ from r to **k**
  keep only k largest singular values

  Σ_K is M×N diagonal matrix: $\Sigma(i,i) = \sqrt{\lambda_i}$ for $1 \le i \le$ **k**
  $\Sigma(i,j) = 0$ otherwise



Reduced Rank Approximation of C

- Approximation:

$$C_k = U\Sigma_k V^T$$

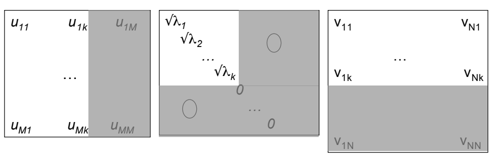
[M×N] [M×M] [M×N] [N×N]

- Theorem:

  C_k is the best rank-k approximation to C under the least square fit (Frobenius) norm

$$= \sqrt{\Sigma^M_{i=1} \Sigma^N_{j=1} (C(i,j) - C_k(i,j))^2}$$

Reduced dimension matrices



C_k =	U'_k	Σ'_k	V'^T_k
M×N	M×k	k×k	k×N

Using the Approximation

- View V'^T_k as a representation of documents in a k-dimensional space
  – a "concept space"?
- Transform query vector **q** into that space:

  $C_k^T C_k = (U'_k \Sigma'_k V'^T_k)^T (U'_k \Sigma'_k V'^T_k) = (V'_k \Sigma'^T_k U'^T_k)(U'_k \Sigma'_k V'^T_k)$
  $= V'_k (\Sigma'_k)^2 (V'_k)^T$ compares documents

  => $C_k^T \boldsymbol{q}$ should = $V'_k (\Sigma'_k)^2 \boldsymbol{q}_k$ compare doc. to query

  => $\boldsymbol{q}_k = (\Sigma'^{-1}_k)^2 V'^T_k C_k^T \boldsymbol{q} = (\Sigma'^{-1}_k)^2 V'^T_k V'_k \Sigma'^T_k U'^T_k \mathbf{q}$
  $= (\Sigma'_k)^{-1} (U'_k)^T \boldsymbol{q}$

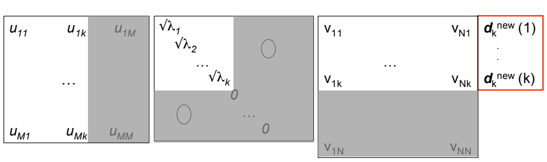
  recalling $(V'^T_k)(V'_k) = (U'^T_k)(U'_k) = I$

Adding a new document

add new document $\boldsymbol{d}^{new}$ to C_k => add column $\boldsymbol{d}_k^{new}$ to V'^T_k

Transform $\mathbf{d}^{new}$ into the k-dimensional space version $\boldsymbol{d}_k^{new}$

$V'^T_k = (\Sigma'_k)^{-1} (U'_k)^T C_k$ => $(\Sigma'_k)^{-1} (U'_k)^T \boldsymbol{d}^{new} = \boldsymbol{d}_k^{new}$



C_k =	U'_k	Σ'_k	V'^T_k
M×(N+1)	M×k	k×k	k×(N+1)

Original LSI paper:

Deerwester, Dumais, et. al.
Indexing by Latent Semantic Analysis
Journal of the Society for Information Science, 41(6), 1990, 391-407.

Example from that paper follows

Deerwester, Dumais et. al. Table:

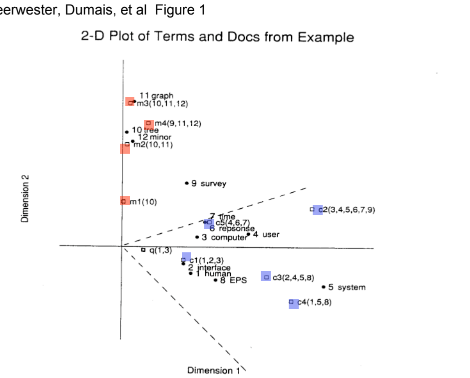
Terms	Documents								
	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph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

Deerwester, Dumais et. al. example, cont.:

Matrix V'^T_k for k=2

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

Deerwester, Dumais, et al Figure 1



Summary

- LSI uses SVD to get a reduced-rank and reduced-size approximation to C
- LSI can be viewed as a preprocessor for
  - query evaluation
  - clustering
- SVD computation can be costly
  - do once (or rarely)

Another application of SVD: collaborative filtering

Modeling Relationships at Multiple Scales to Improve Accuracy of Large Recommender Systems, Robert M. Bell, Yehuda Koren and Chris Volinsky, *KDD 07*

•one of methods used for Netflix challenge

•find factors to describe user and items

•use to fill in vaues of unknown ratings of items by users