Chapter 4:
Information Retrieval Models

Overview

- Definitions
- Important IR models:
  - based on term matching
    - set theoretic
    - algebraic
    - probabilistic
  - based on link analysis [see Web Information Retrieval]

Definitions

- **Information retrieval models** (also called ranking or relevance models)
  - defined by :
    - the form used in representing document text and query
    - by the ranking procedure
  - often incorporate:
    - element of uncertainty

Formal characterization of IR models

A retrieval model is a quadruple $\langle D, Q, F, R(d_j, q_i)\rangle$ where:

1. D is a set composed of representations of the documents in the collection
2. Q is a set composed of representations of the queries of a user
3. F is a framework for modelling document and query representations, and their relationships
4. $R(q_i, d_j)$ is a ranking function:
    - takes into account document representation $d_j \in D$ and a query representation $q_i \in Q$
    - associates a real number that expresses the potential relevance of d_j to q_i by which documents can be ordered

Basic concepts

In this course unit retrieval models will be mainly illustrated with representations composed of index terms

- t = number of index (vocabulary) terms in the collection
- $K = \{k_1,...,k_t\}$ = set of all index terms
- A weight $w_{ij} > 0$ is associated with each index term k_i of a document d_j or query q
- For a k_i that does not appear in the document or query text: respectively w_{ij} or $w_{iq} = 0$
- $[w_{1j}, w_{2j}, ..., w_{tj}]$: term vector of d_j
- $[w_{1q}, w_{2q}, ..., w_{tq}]$: term vector of q
- $g_i(d_j) = w_{ij}$: g_i is a function that returns the weight associated with the index term k_i (here of d_j)

- Term representation: **OVERSIMPLIFICATION** of the semantics of documents and query
  - increasingly: semantics are added to documents or query
  - e.g., latent semantic topic models, structural metadata of the document, semantic labels of query and document obtained with information extraction technology, contextual information, tags of users, ...

Taxonomy of important retrieval models

Term-based

1) **Set theoretic**:

- documents and queries: represented as a set of index terms
  - Boolean model
  - extended Boolean model

2) **Algebraic**:

- documents and queries represented as vectors in a *t*-dimensional space (*t* = number of index terms in the collection)
  - vector space model
  - latent semantic indexing model

Taxonomy of important retrieval models

3) **Probabilistic**:

- framework for modeling document and query based on probabilistic theory
  - classic probabilistic model
  - **language model**
  - inference network model

- **Link-based**: see [Web Information Retrieval]

Boolean model

- The index term weight variables are all binary:
  $w_{ij} \in \{0, 1\}$ and $w_{iq} \in \{0, 1\}$
- A query q is a conventional Boolean expression:
  - index terms connected with Boolean operators ($\wedge$ $\vee$ $\neg$)
- Let q_{dnf} be the disjunctive normal form for the query q
- Let q_{cc} be any of the conjunctive components of q_{dnf}
- The **similarity** of a document d_j to the query q is defined as:

$$sim(d_j, q) = \begin{cases} 1 & if\ \exists q_{cc} \mid (q_{cc} \in q_{dnf}) \wedge (\forall_{k_i}, g_i(d_j) = g_i(q_{cc})) \\ 0 & \text{otherwise} \end{cases}$$

Boolean model: example

set of index terms=

{lawyer, car, blue, theft, family}

k_1 k_2 k_3 k_4 k_5

Doc1: describes a family buying a blue car: (0,1,1,0,1)

Doc2: describes a blue car theft defended by a lawyer: (1,1,1,1,0)

Doc3: describes a family asking a lawyer's advice: (1,0,0,0,1)

Query = $(\neg \text{ blue } \vee \neg \text{ lawyer}) \wedge \text{car} \wedge \text{theft}$

DNF (disjunctive normal form) of query = (0,1,0,1,0) $\vee$ (1,1,0,1,0) $\vee$ (0,1,0,1,1) $\vee$ (1,1,0,1,1) $\vee$ (0,1,1,1,1) $\vee$ (0,1,1,1,0)

Which documents are relevant?

Boolean model

- Advantage:
  - simplicity: (was) popular in commercial systems

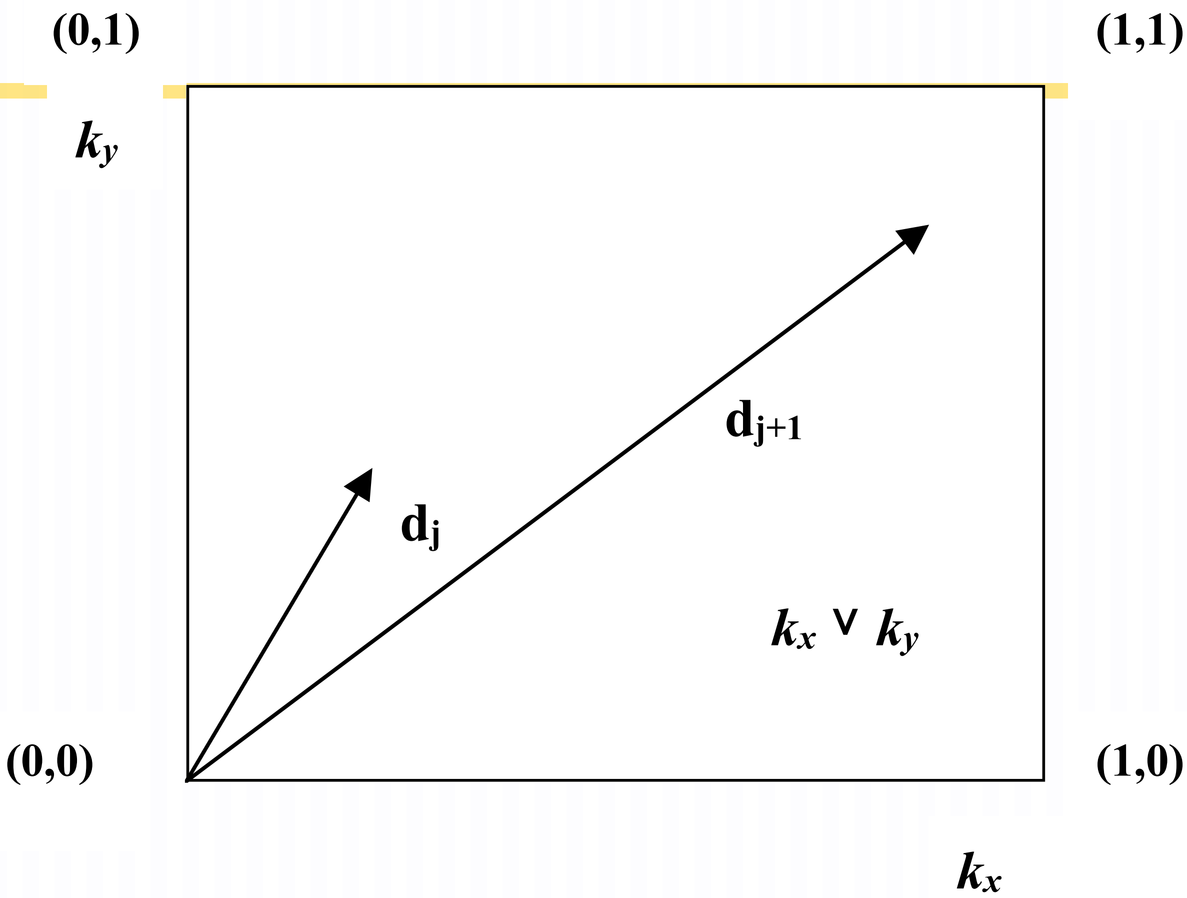
- Disadvantages:
  - difficult for the user to express information need as a Boolean expression
  - pure model:
    - relative importance of index terms ignored
    - no ranking (document = relevant or non-relevant)
  - variant models take into account partial fulfillment of the Boolean query: e.g., **extended Boolean model**

Extended Boolean model

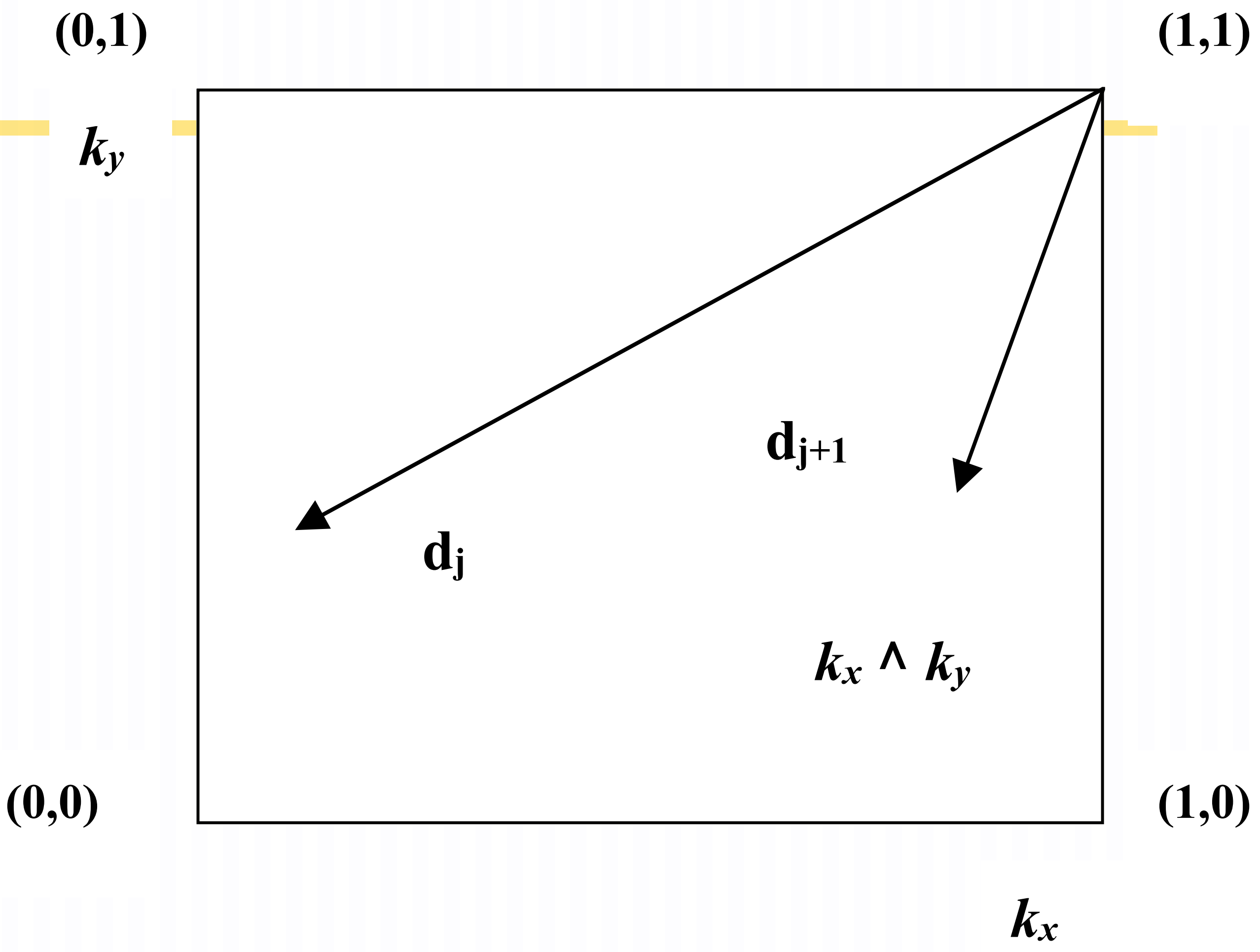
- Illustrated with small example:
  - consider a two-dimensional space with index terms k_x and k_y in which documents and queries can be plotted
  - document d_j is positioned by weights w_{xj} and w_{yj} both between 0 and 1
  - **for disjunctive query** $q_{or} = k_x \vee k_y$: the point (0,0) = point to be avoided

    => Euclidean distance from (0,0) to d_j : similarity measure for (d_j,q_{or})
  - **for conjunctive query** $q_{and} = k_x \wedge k_y$: the point (1,1) = most desirable spot

    => complement of the Euclidean distance from (1,1) to d_j : similarity measure for (d_j,q_{and})



(0,1)
(1,1)
k_y
d_{j+1}
d_j
$k_x \vee k_y$
(0,0)
(1,0)
k_x


(0,1)
(1,1)
k_y
d_{j+1}
d_j
$k_x \wedge k_y$
(0,0)
(1,0)
k_x

Extended Boolean model

- Distances can be normalized, which yields:

$$sim(d_j, q_{or}) = \left(\frac{w_{xj}^2 + w_{yj}^2}{2} \right)^{\frac{1}{2}}$$

$$sim(d_j, q_{and}) = 1 - \left(\frac{(1-w_{xj})^2 + (1-w_{yj})^2}{2} \right)^{\frac{1}{2}}$$

- Can be generalized over *m* query terms and *p*-distances:

$$sim(d_j, q_{or}) = \left(\frac{w_{1j}^p + w_{2j}^p + \ldots + w_{mj}^p}{m} \right)^{\frac{1}{p}}$$

$$sim(d_j, q_{and}) = 1 - \left(\frac{(1-w_{1j})^p + (1-w_{2j})^p + \ldots + (1-w_{mj})^p}{m} \right)^{\frac{1}{p}}$$

Extended Boolean model

- The processing of more general queries: e.g., $q = (k_1 \wedge k_2) \vee k_3$
- ***p*-norm model**:
  - generalizes notion of distance to include both Euclidean distances and p-distances where $1 \leq p \leq \infty$ dependent upon the collection, but usually chosen as $2 \leq p \leq 5$
  - takes into account partial fulfillment of conjunctive and disjunctive queries
  - hybrid model: properties of set theoretic and algebraic model
  - results: **ranking** of documents

Vector (space) model (VSM)

- **Vector (space) model**: Document and query are represented as term vectors with term weights ≥ 0 in a *t*-dimensional space:

  $\boldsymbol{d_j} = [w_{1j}, w_{2j}, \ldots, w_{tj}]$

  $\boldsymbol{q} = [w_{1q}, w_{2q}, \ldots, w_{tq}]$

  where *t* = the number of features (here terms) measured

Vector (space) model (VSM)

- The distance between the document $\boldsymbol{d}_j$ and the query $\boldsymbol{q}$ is defined e.g. as:
  - Manhattan distance
  - Euclidean distance
- The similarity of a document $\boldsymbol{d}_j$ and the query $\boldsymbol{q}$ is defined e.g. as:
  - inner product similarity
  - **cosine similarity** (most popular)
  - Dice similarity
- Result: **ranking** of documents

- Manhattan distance:

$$dis(\boldsymbol{d}_j, \boldsymbol{q}) = \sum_{i=1}^{t} \left| w_{ij} - w_{iq} \right|$$

- Euclidean distance:

$$dis(\boldsymbol{d}_j, \boldsymbol{q}) = \sqrt{\sum_{i=1}^{t} (w_{ij} - w_{iq})^2}$$

- Inner product similarity:

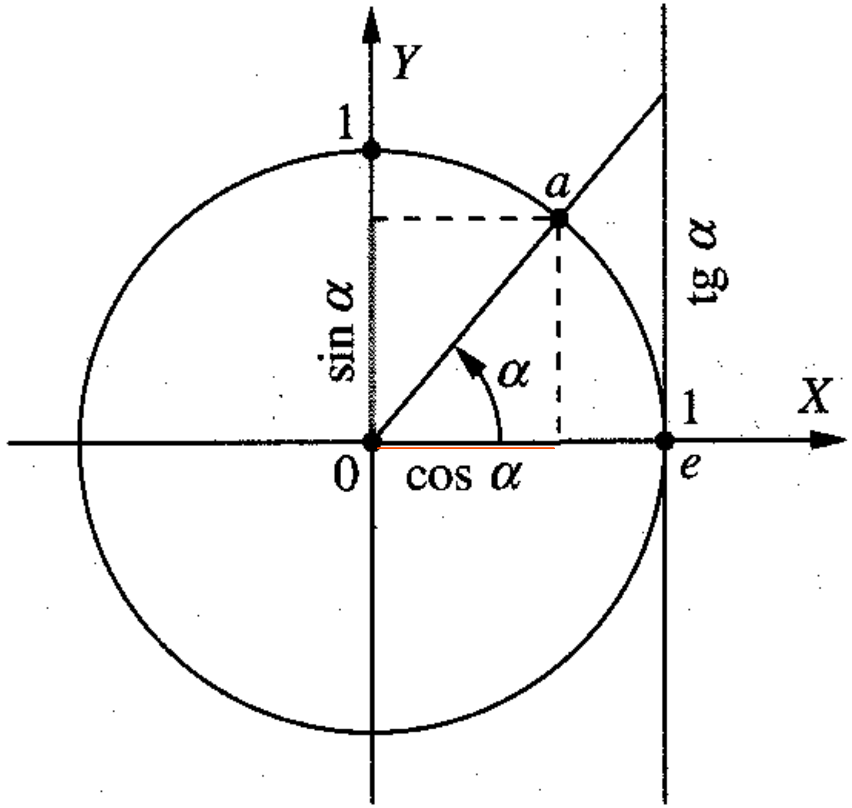
$$sim(\boldsymbol{d}_j,\boldsymbol{q}) = \boldsymbol{d}_j^{\,T} \cdot \boldsymbol{q} = \sum_{i=1}^{t} w_{ij} w_{iq}$$

- **Cosine similarity**: cosine of the angle between document and query vectors:

$$sim(\boldsymbol{d}_j,\boldsymbol{q}) = \frac{\boldsymbol{d}_j^{\,T} \cdot \boldsymbol{q}}{\|\boldsymbol{d}_j\| \, \|\boldsymbol{q}\|} = \frac{\sum_{i=1}^{t} w_{ij} w_{iq}}{\sqrt{\sum_{i=1}^{t} w_{ij}^2} \sqrt{\sum_{i=1}^{t} w_{iq}^2}}$$

- Dice similarity:

$$sim(\boldsymbol{d}_j, \boldsymbol{q}) = \frac{2\sum_{i=1}^{t} w_{ij} w_{iq}}{\sum_{i=1}^{t} w_{ij} + \sum_{i=1}^{t} w_{iq}}$$


Y
1
a
sin α
tg α
α
1
X
0
cos α
e

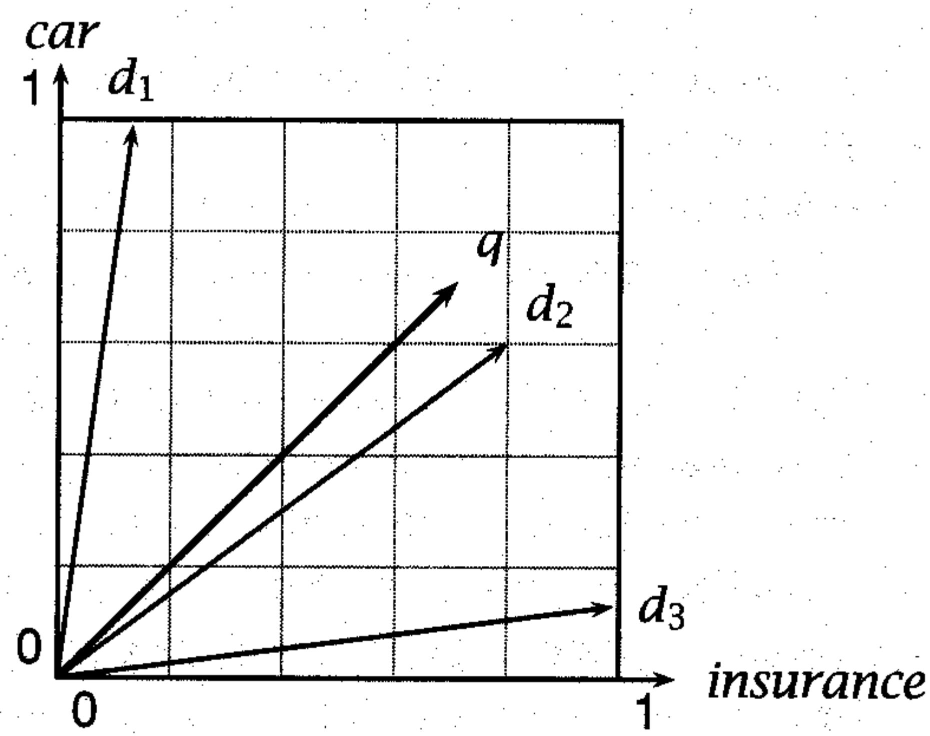


Figure 15.3 A vector space with two dimensions. The two dimensions correspond to the terms *car* and *insurance*. One query and three documents are represented in the space.

Source: Manning & Schütze, 2000

Vector space model

- Advantages:
  - term weighting scheme improves performance
  - partial matching: retrieval of documents that approximate the query conditions
  - simple, efficient model with relatively good results: popular (e.g., SMART system)
  - easy re-weighting of query terms in relevance feedback
- Disadvantage:
  - simplifying assumption that terms are not correlated and term vectors are pair-wise orthogonal

Latent semantic indexing (LSI)

- **Latent semantic indexing** (LSI) or **Latent semantic analysis**
  - can be incorporated in information retrieval model as variant of the vector space model:
    - mapping the term vectors of documents and query into a low dimensional space which is associated with statistical **concepts**
    - this would allow the retrieval of documents even when they are not indexed by the query index terms
- LSI = method for dimensionality reduction using method of Singular Value Decomposition (SVD) [Deerweester et al. 1990]

Singular Value Decomposition example

Given the 4 x 2 matrix A

$$A = \begin{bmatrix} 1 & 1 \\ 2 & -2 \\ -2 & 2 \\ -1 & -1 \end{bmatrix}$$

Compute singular value decomposition: $A = U \sum V^T$

Compute $A^T A$

$$A^T A = \begin{bmatrix} 1 & 2 & -2 & -1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 2 & -2 \\ -2 & 2 \\ -1 & -1 \end{bmatrix} = \begin{bmatrix} 10 & -6 \\ -6 & 10 \end{bmatrix}$$

Singular Value Decomposition example

This matrix has two eigenvalues. Sort them from large to small.

$\lambda_1 = 16$ $\lambda_2 = 4$

Search for each eigenvalue the eigenvector.

$$\lambda_1 = 16: \begin{bmatrix} -6 & -6 \\ -6 & -6 \end{bmatrix} V_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow \mathrm{V}_1 = \begin{bmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{bmatrix}$$

$$\lambda_2 = 4: \begin{bmatrix} 6 & -6 \\ -6 & 6 \end{bmatrix} V_2 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow \mathrm{V}_2 = \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}$$

The orthogonal matrix V with in its columns V_1 and V_2 are thus found.

Singular Value Decomposition example

The singular values are: $\sigma_1 = \sqrt{\lambda_1} = 4$ $\sigma_2 = \sqrt{\lambda_2} = 2$

So matrix $\sum$ is also computed.

Computations matrix U based on AA^T cf. above.

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

$$\begin{bmatrix} 1 & 1 \\ 2 & -2 \\ -2 & 2 \\ -1 & -1 \end{bmatrix} = \begin{bmatrix} 0 & 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 1/\sqrt{2} & 0 & 0 & 1/\sqrt{2} \\ -1/\sqrt{2} & 0 & 0 & 1/\sqrt{2} \\ 0 & -1/\sqrt{2} & 1/\sqrt{2} & 0 \end{bmatrix} \begin{bmatrix} 4 & 0 \\ 0 & 2 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}$$

Latent semantic indexing

- = application of Singular Value Decomposition (SVD) to a term-document matrix A_{txm}

  where

  t = number of index terms in the collection

  m = the number of unique documents in the collection

Latent semantic indexing

- A is decomposed with SVD as the product of 3 matrices:

  $$A = U \sum V^T \quad \text{where}$$

  U_{txt} = orthogonal matrix of eigenvectors (left singular vectors) derived from the term-to-term correlation matrix given by AA^T

  V_{mxm}= orthogonal matrix of eigenvectors (right singular vectors) derived from the document-to-document correlation matrix given by A^TA

  $\sum$ = diagonal matrix with the singular values of A: indicates the importance of the corresponding singular vectors in matrices U and V

Latent semantic indexing

- Reduced vector space:
  - Select *k* dimensions as basis for the reduced vector space
  - $k < t$ and $k < m$ (k = usually 50-300)
  - singular values are ordered by size: keep the k largest values and keep the corresponding columns from the U and V matrices: U_k and V_k
- The product of the resulting matrices:

$$U_k \Sigma_k V_k^T = \hat{A}$$

- $\hat{A}$ represents A as in a lower k dimensional space (rank k): in such a way that the representations in the original space are changed as little as possible when measured by the sum of squares of the differences

Latent semantic indexing

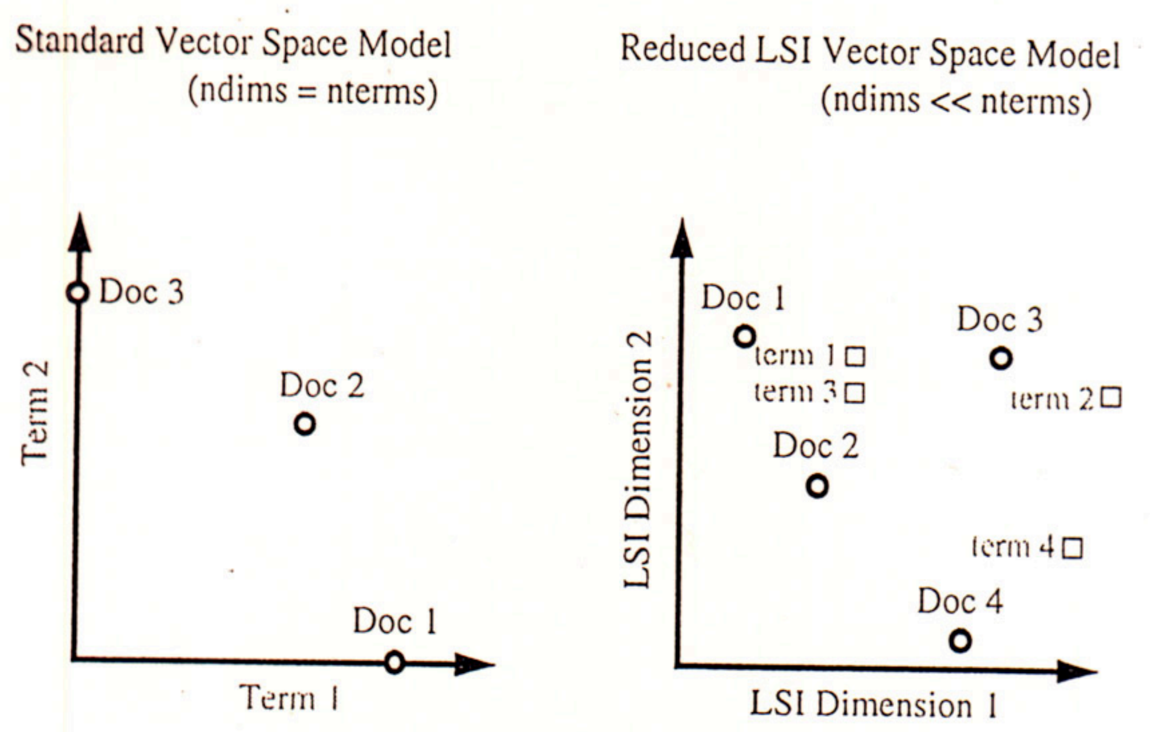


Figure 1 Term representations in the standard vector vs. reduced LSI vector models.

[Grefenstette 1998]

Latent semantic indexing

- The LSI approach makes 3 claims:
  - the semantic information can be derived from a word-document co-occurrence matrix
  - dimensionality reduction is an essential part of this derivation
  - the words and documents can be represented as points in the Euclidean space

Latent semantic indexing

- Advantage:
  - more compact representations of documents
- Disadvantages:
  - Number of topic dimensions?
  - Cannot handle negation, specific queries (e.g., names of companies)

- Integrated in commercial systems

LSI retrieval model

- Query and (usually) documents are mapped to the lower dimensional space
- In reduced vector space: term in U_k is represented as linear combination of documents:
  - "synonyms grouped"
- L contains weighted singular vectors in U_k, where the weights are the corresponding singular values in $\sum_k$:

$$L = U_k \sum_k^{-1}$$

-1: inverted values of the singular values

- The **similarity** *sim* ($\boldsymbol{d}_j$, $\boldsymbol{q}$) of the document $\boldsymbol{d}_j$ to the query $\boldsymbol{q}$ is often defined as:

$$sim(\boldsymbol{d}_j, q) = \cos(L^T \boldsymbol{d}_j, L^T \boldsymbol{q})$$

Probabilistic retrieval model

- **Probabilistic retrieval model**: Views retrieval as a problem of estimating the probability of relevance given a query, document, collection, ...
- Aims at **ranking** the retrieved documents in decreasing order of this probability
- Many different models:
  - generative relevance models:
    - classic probabilistic model
    - language model
  - inference network model

Generative relevance models

- Random variables:
  - D = document
  - Q = query
  - R = relevance: $R = r$ (relevant) or $R = \bar{r}$ (not relevant)
- Basic question:
  - estimating:

$$P(R = r \mid D, Q) = 1 - P(R = \bar{r} \mid D, Q)$$

Generative relevance models

- Generative relevance model: $P(R = r \mid D, Q)$

  is not estimated directly, but is estimated indirectly via Bayes' rule:

$$P(R = r \mid D, Q) = \frac{P(D, Q \mid R = r)P(R = r)}{P(D, Q)}$$

  equivalently, we may use the log-odds to rank documents:

$$\log \frac{P(R = r \mid D, Q)}{P(R = \bar{r} \mid D, Q)} = \log \frac{P(D, Q \mid R = r)P(R = r)}{P(D, Q \mid R = \bar{r})P(R = \bar{r})}$$

Bayes' rule

$$P(B \mid A) = \frac{P(B \cap A)}{P(A)} = \frac{P(A \mid B)P(B)}{P(A)}$$

allows calculating $P(B|A)$ in terms of $P(A|B)$ when the former quantity is difficult to determine

Classic probabilistic model = binary independence retrieval model

- $P(D,Q|R)$ is factored as $P(D,Q|R) = P(Q|R)P(D|Q,R)$ by applying the chain rule leading to the following log-odds ratios:

$$\log \frac{P(R=r|D,Q)}{P(R=\bar{r}|D,Q)} = \log \frac{P(D,Q\mid R=r)P(R=r)}{P(D,Q\mid R=\bar{r})P(R=\bar{r})}$$

$$= \log \frac{P(D\mid Q,R=r)P(Q|R=r)P(R=r)}{P(D\mid Q,R=\bar{r})P(Q|R=\bar{r})P(R=\bar{r})}$$

Bayes'rule and removal of terms for the purpose of ranking

[Robertson & Sparck Jones 1976]

Classic probabilistic model

$$= \log \frac{P(D \mid Q, R = r) P(R = r \mid Q)}{P(D \mid Q, R = \bar{r}) P(R = \bar{r} \mid Q)}$$

$$= \log \frac{P(D \mid Q, R = r)}{P(D \mid Q, R = \bar{r})} + \log \frac{P(R = r \mid Q)}{P(R = \bar{r} \mid Q)}$$

The latter term can be safely removed for the purpose of ranking

$$\overset{rank}{=} \log \frac{P(D \mid Q, R = r)}{P(D \mid Q, R = \bar{r})}$$

Classic probabilistic model

- Assuming that the document is made up of a collection of attributes:

  $D = (W_1, \ldots, W_n)$, such as words, and that these attributes are independent given R and Q:

$$P(D|Q, R = r) = \prod_{i=1}^{n} P(W_i|Q, R = r)$$

$$P(D|Q, R = \bar{r}) = \prod_{i=1}^{n} P(W_i|Q, R = \bar{r})$$

Classic probabilistic model

- Learns the properties of the sets of relevant and non-relevant documents:
  - initial query: guessing these properties or guessing $P(W_i|Q,R=r)$ and $P(W_i|Q,R=\bar{r})$
  - e.g., $P(W_i|Q,R=r)=0.5$ $P(W_i|Q,R=\bar{r}) = \frac{n_i}{N}$

    where n_i = number of documents which contain w_i
    N = number of documents in the collection
  - by relevance feedback: better estimation of the probabilities
  - retrieval can be iterated with new probability estimates until user is satisfied

Language model

[Lafferty & Zhai 2003]

- $P(D,Q|R)$ is factored as $P(D,Q|R) = P(D|R)P(Q|D,R)$ by applying the chain rule leading to the following log-odds ratios:

$$\log \frac{P(R=r|Q,D)}{P(R=\bar{r}|Q,D)} = \log \frac{P(Q,D \mid R=r)P(R=r)}{P(Q,D \mid R=\bar{r})P(R=\bar{r})}$$

$$= \log \frac{P(Q \mid D,R=r)P(D|R=r)P(R=r)}{P(Q \mid D,R=\bar{r})P(D|R=\bar{r})P(R=\bar{r})}$$

Bayes'rule and removal of terms for the purpose of ranking

Language model

$$= \log \frac{P(Q \mid D, R = r) P(R = r \mid D)}{P(Q \mid D, R = \bar{r}) P(R = \bar{r} \mid D)}$$

$$= \log \frac{P(Q \mid D, R = r)}{P(Q \mid D, R = \bar{r})} + \log \frac{P(R = r \mid D)}{P(R = \bar{r} \mid D)}$$

The latter term is dependent on D, but independent on Q, thus can be considered for the purpose of ranking.

Assume that conditioned on the event $R = \bar{r}$, the document D is independent of the query Q , i.e.,

$$P(D, Q \mid R = \bar{r}) = P(D \mid R = \bar{r}) P(Q \mid R = \bar{r})$$

Language model

$$\log \frac{P(R=r|Q,D)}{P(R=\bar{r}|Q,D)} = \log \frac{P(Q \mid D, R=r)}{P(Q \mid R=\bar{r})} + \log \frac{P(R=r|D)}{P(R=\bar{r}|D)}$$

$$\stackrel{rank}{=} \log P(Q|D,R=r) + \log \frac{P(R=r \mid D)}{P(R=\bar{r} \mid D)}$$

Assume that D and R are independent, i.e.,

$$P(D,R) = P(D)P(R)$$

$$\stackrel{rank}{=} \log P(Q|D,R=r)$$

Language model

- Each query is made of *m* attributes (e.g., n-grams): $Q = (Q_1,\ldots,Q_m)$, typically the query terms, assuming that the attributes are independent given the document and *R*:

$$\log \frac{P(R=r|Q,D)}{P(R=\bar{r}|Q,D)} \stackrel{rank}{=} \log \prod_{i=1}^{m} P(Q_i|D,R=r) + \log \frac{P(R=r\mid D)}{P(R=\bar{r}\mid D)}$$

$$\stackrel{rank}{=} \sum_{i=1}^{m} \log P(Q_i|D,R=r) + \log \frac{P(R=r\mid D)}{P(R=\bar{r}\mid D)}$$

- Strictly LM assumes that there is just one document that generates the query and that the user knows (or correctly guesses) something about this document

Language model

- A language retrieval model **ranks** a document (or information object) *D* according to the probability that the document generates the query (i.e., $P(Q|D)$)
- Suppose the query Q is composed of *m* query terms q_i:

$$P(q_1,\ldots,q_m|D) = \prod_{i=1}^{m}(\lambda P_{ML}(q_i|D) + (1-\lambda)P_{ML}(q_i|C))$$

where *C* = document collection

λ = Jelinek-Mercer smoothing parameter

The simplest estimation is by by maximum likelihood (ML):

$$P_{ML}(q_i|D) = \frac{f(q_i,D)}{|D|} \qquad P_{ML}(q_i|C) = \frac{f(q_i,C)}{|C|}$$

Common smoothing methods used in IR

- Smoothing of the probabilities = reevaluating the probabilities: assign some non-zero probability to query terms that do not occur in the document
- **Jelinek-Mercer smoothing**:

$$P(q_i|D) = \lambda P_{ML}(q_i|D) + (1-\lambda)P_{ML}(q_i|C))$$

  where $\lambda \in [0,1]$

- **Dirichlet smoothing**: $P(q_i|D) = \dfrac{f(q_i,D) + \mu P_{ML}(q_i|C)}{|D| + \mu}$

  where μ = Dirichlet prior

Language model

- Value of λ is obtained from a sample collection:
- set empirically
- estimated by the EM (expectation maximization) algorithm
- often for each query term a λ_i is estimated denoting the importance of each query term, e.g. with the EM algorithm and relevance feedback

Language model

- The EM-algorithm iteratively maximizes the probability of the query given *r* relevant documents $Rd_1,\ldots,Rd_r$:

  init $\lambda_i^{(0)}$ (e.g.: 0.5)

  E-step: $$m_i = \sum_{j=1}^{r} \frac{\lambda_i^{(p)} P(q_i \mid Rd_j)}{(1-\lambda_i^{(p)})P(q_i \mid C) + \lambda_i^{(p)} P(q_i \mid Rd_j)}$$

  M-step: $$\lambda_i^{(p+1)} = \frac{m_i}{r}$$

  Each iteration *p* estimates a new value $\lambda_i^{(p+1)}$ by first computing the E-step and then the M-step until the value $\lambda_i^{(p+1)}$ is not anymore significantly different from $\lambda_i^{(p)}$

Language model

- Allows integrating the translation of a certain content pattern into a conceptual term and the probability of this translation:

$$P(cq_1,\ldots,cq_m|D) = \prod_{i=1}^{m}(\alpha\sum_{l=1}^{k}P(cq_i|w_l)P(w_l|D) + \beta P(cq_i|D) + (1-\alpha-\beta)P(cq_i|C))$$

where cq_i = conceptual terms

w_l = content pattern (e.g., word, image pattern)

Adding a language model for the query

- **Language model for the query** (e.g., based on relevant documents, maximum likelihood of the query terms, or on concepts of a user's profile) [see Cross Language IR]

Adding a language model for the query

- Let θ_Q and θ_D be the language model of the query *Q* and document *D* respectively
- Relevance is computed based on the divergence of two language models (where *w* is a word of the vocabulary):
  - **Kullback-Leibler divergence** or relative entropy:

$$KL(\theta_Q \| \theta_D) = \sum_w P(w|Q) \log \frac{P(w|Q)}{P(w|D)}$$

  - **cross-entropy**:

$$H(\theta_Q \| \theta_D) = -\sum_w P(w|Q) \log P(w|D)$$

- Documents are ranked by increasing divergence

Inference network model

- Example of the use of a **Bayesian network** in retrieval
- = directed acyclic graph (DAG)
  - **nodes** = random variables
  - **arcs** = causal relationships between these variables
    - causal relationship is represented by the edge $e = (u,v)$ directed from each parent (tail) node u to the child (head) node v
    - parents of a node are judged to be direct causes for it
    - strength of causal influences are expressed by **conditional probabilities**
  - **roots** = nodes without parents
    - have **prior probability**: e.g., given based on domain knowledge

Inference network model

- **Document network (DAG)**:
  - contains document representations
  - document (e.g., d_i) represented by:
    - text nodes (e.g., t_j), concept nodes (e.g., r_k), other representation nodes (e.g., representing figures, images) and relations
  - often a document network is once built for the complete document collection:
    - prior probability of a document node

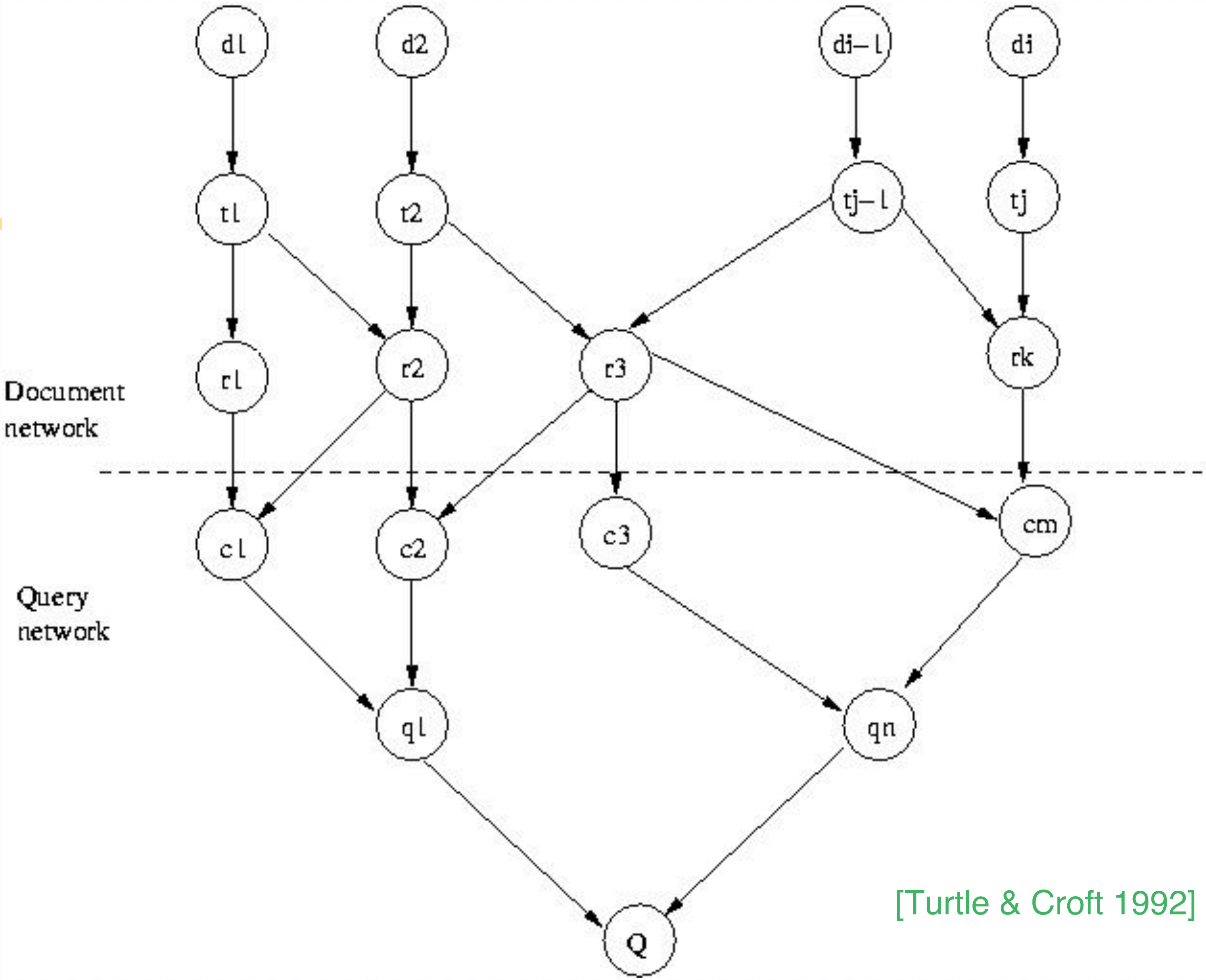
Inference network model

- **Query network (inverted DAG)**:
  - single leaf: information need (*Q*)
  - information need can have different representations (e.g., q_i) e.g., made up of terms or concepts (e.g., c_l)
  - a query representation can be represented by concepts
- **Retrieval**:
  - the two networks are connected e.g., by their common terms or concepts (attachment) to form the inference or causal network
  - retrieval = a process of combining uncertain evidences from the network and inferring a probability or belief that a document is relevant

Inference network model

- for each document instantiated (e.g. d_j = true (=1), while remaining documents are false (= 0)): the conditional probability for each node in the network is computed
- probability is computed as the propagation of the probabilities from a document node d_j to the query node q

- several evidence combination methods for computing the conditional probability at a node given the parents:
  - e.g., to fit the normal Boolean logic
  - e.g. (weighted) sum: belief a node computed as (weighted) average probability of the parents
- documents are **ranked** according to their probability of relevance



[Turtle & Croft 1992]

Operators supported by the INQUERY system (University of Massachusetts Amherst, USA) :

#and : AND the terms
#or: OR the terms
#not: negate the term (incoming belief)
#sum: sum of the incoming beliefs
#wsum: weighted sum of the incoming beliefs
#max: maximum of the incoming beliefs

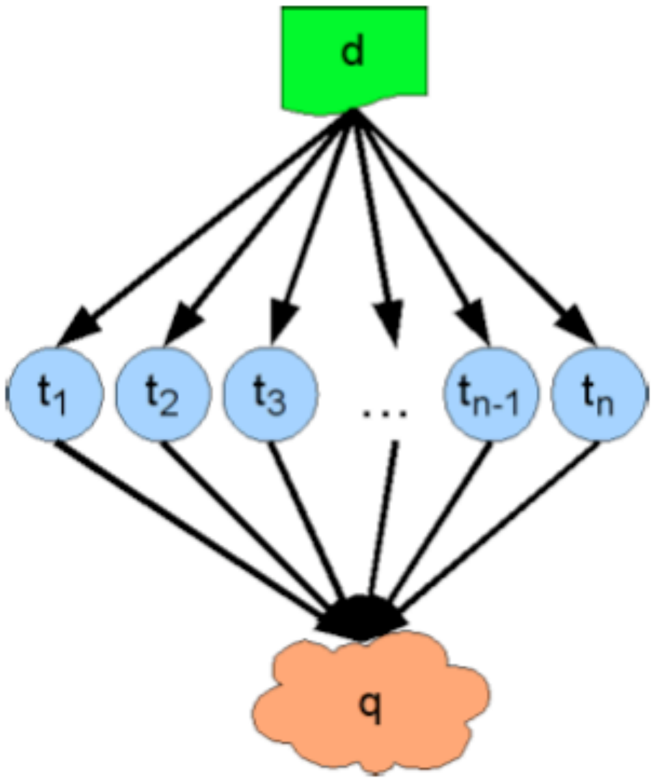
See exercises for examples

Simplified form: cf. vector space model

- $$P(d \to q) = \sum_t P(d \to t)P(t \to q)$$

  $P(d \to t)$: representation of document d
  $\widehat{=}$ document indexing

  $P(t \to q)$: representation of query q
  $\widehat{=}$ query formulation



$$P(d \to q) \approx \sum_t P(d \to t)P(t \to q) = \vec{d} \cdot \vec{q}$$

Inference network model

- Advantages:
  - combines multiple sources of evidence and probabilistic dependencies in a very elegant way to suit the general probabilistic paradigm:

    P(Query | Document, Document representation, Collection representation, External knowledge,…)
  - easy integration of representations of different media, domain knowledge, semantic information, ...
  - good retrieval performance with general collections
- **Much new Bayesian network technology yet to be applied !**

What have we learned?

- Commercial information retrieval systems:
  - Boolean and vector space model: widespread
  - increasingly incorporation of latent semantic topic models
  - **language models and inference net models: most potential to model document and information need**
- Important:
  - **to understand how the models handle problems of uncertain representations, partial matching, term correlation, and expansion of terms with "synonyms"**
  - **to understand how to incorporate relevance feedback in the models**

Research questions to be solved

- Further investigations into probabilistic content models of retrievable objects
- Approximate inference in retrieval models (e.g., Bayesian networks)

Further reading

Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K. & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science,* 41 (6), 391-407.

Grefenstette, G. (1998). *Cross-language Information Retrieval*. Kluwer Academic Publishers.

Lafferty, J. & Zhai C. X. (2003). Probabilistic relevance models based on document and query generation. In W.B. Croft & J. Lafferty (Eds.), *Language Modeling for Information Retrieval* (pp. 1-10). Boston: Kluwer Academic Publishers.

Rocchio, J.J. (1971). Relevance feedback in information retrieval. In G. Salton (Ed.), *The SMART Retrieval System: Experiments in Automatic Document Processing* (pp. 313-323). Englewood Cliffs, NJ: Prentice Hall.

Robertson, S. & Sparck Jones, K. (1976). Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27, 129-146.

Turtle, H.R. & Croft, W.B. (1992). A comparison of text retrieval models. *The Computer Journal,* 35 (3), 279-290.

Language modeling toolkit for IR: http://www-2.cs.cmu.edu/lemur/