Models for Retrieval

1. HMM/N-gram-based
2. Latent Semantic Indexing (LSI)
3. Probabilistic Latent Semantic Analysis (PLSA)

Berlin Chen 2003

References:

1. Berlin Chen et al., “An HMM/N-gram-based Linguistic Processing Approach for Mandarin Spoken Document Retrieval,” EUROSPEECH 2001
2. M. W. Berry et al., “Using Linear Algebra for Intelligent Information Retrieval,” technical report, 1994
3. Thomas Hofmann, “Unsupervised Learning by Probabilistic Latent Semantic Analysis,” Machine Learning, 2001

HMM/N-gram-based Model

- Model the query Q as a sequence of input observations (index terms), $Q = q_1 q_2 .. q_n .. q_N$
- Model the doc D as a discrete HMM composed of distribution of *N*-gram parameters
- The relevance measure, $P(Q|D \text{ is } R)$, can be estimated by the *N*-gram probabilities of the index term sequence for the query, $Q = q_1 q_2 .. q_n .. q_N$, predicted by the doc D
  - A generative model for IR

$$
\begin{aligned}
D^* &= \arg\max_D P(D \text{ is } R | Q) \\
&\approx \arg\max_D P(Q|D \text{ is } R) \cancel{P(D \text{ is } R)} \\
&\approx \arg\max_D P(Q|D \text{ is } R)
\end{aligned}
$$

with the assumption that ……

HMM/N-gram-based Model

$$P(W) \quad \{W = w_1 w_2 .. w_n .. w_N\}$$
$$= P(w_1 w_2 .. w_n .. w_N)$$
$$= P(w_1)P(w_2|w_1)P(w_3|w_1 w_2) .... P(w_N|w_1 w_2 .... w_{N-1})$$

- *N*-gram approximation (Language Model)
  - Unigram

$$P(W) = P(w_1)P(w_2)P(w_3) .... P(w_N)$$

  - Bigram

$$P(W) = P(w_1)P(w_2|w_1)P(w_3|w_2) .... P(w_N|w_{N-1})$$

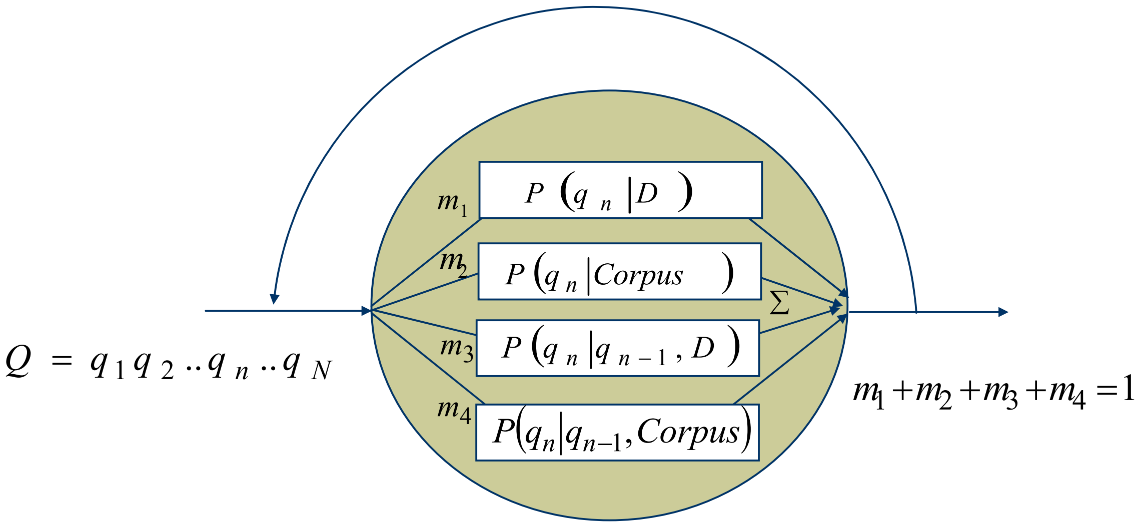
  - Trigram

$$P(W) = P(w_1)P(w_2|w_1)P(w_3|w_1 w_2) .... P(w_N|w_{N-2} w_{N-1})$$

  - ……..

HMM/N-gram-based Model

- A discrete HMM composed of distribution of *N*-gram parameters



$$P(Q|D \text{ is } R) = \left[m_1 P(q_1|D) + m_2 P(q_1|Corpus)\right]$$

$$\cdot \prod_{n=2}^{N} \left[m_1 P(q_n|D) + m_2 P(q_n|Corpus) + m_3 P(q_n|q_{n-1}, D) + m_4 P(q_n|q_{n-1}, Corpus)\right]$$

HMM/N-gram-based Model

- Three Types of HMM Structures
  - Type I: Unigram-Based (Uni)

$$P(Q|D \text{ is } R) = \prod_{n=1}^{N} \left[m_1 P(q_n|D) + m_2 P(q_n|Corpus) \right]$$

  - Type II: Unigram/Bigram-Based (Uni+Bi)

$$P(Q|D \text{ is } R) = \left[m_1 P(q_1|D) + m_2 P(q_1|Corpus) \right] \cdot \prod_{n=2}^{N} \left[m_1 P(q_n|D) + m_2 P(q_n|Corpus) + m_3 P(q_n|q_{n-1}, D) \right]$$

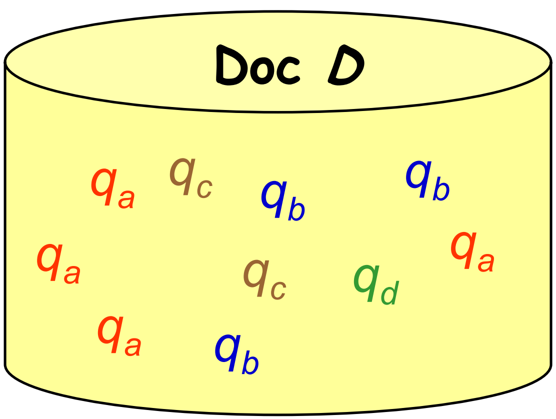
  - Type III: Unigram/Bigram/Corpus-Based (Uni+Bi*)

$$P(Q|D \text{ is } R) = \left[m_1 P(q_1|D) + m_2 P(q_1|Corpus) \right] \cdot \prod_{n=2}^{N} \left[m_1 P(q_n|D) + m_2 P(q_n|Corpus) + m_3 P(q_n|q_{n-1}, D) + m_4 P(q_n|q_{n-1}, Corpus) \right]$$

P(陳水扁 總統 視察 阿里山 小火車|D)
=[m_1P(陳水扁|D)+m_2 P(陳水扁|C)] x [m_1P(總統|D)+m_2P(總統|C)+ m_3P(總統|陳水扁, D)+m_4P(總統|陳水扁, C)]
x[m_1P(視察|D)+m_2P(視察|C)+ m_3P(視察|總統, D)+m_4P(視察|總統, C)] x ……….

HMM/N-gram-based Model

- The role of the corpus *N*-gram probabilities $P(q_n|Corpus)$ $P(q_n|q_{n-1}, Corpus)$
  - Model the general distribution of the index terms
    - Help to solve zero-frequency problem $P(q_n|D) = 0$!
    - Help to differentiate the contributions of different missing terms in a doc
  - The corpus *N*-gram probabilities were estimated using an outside corpus



$P(q_a|D)=0.4$

$P(q_b|D)=0.3$

$P(q_c|D)=0.2$

$P(q_d|D)=0.1$

$P(q_e|D)=0.0$

$P(q_f|D)=0.0$

HMM/N-gram-based Model

- Estimation of *N*-grams (Language Models)
  - Maximum likelihood estimation (MLE) for doc *N*-grams
    - Unigram

$$P(q_i|D)=\frac{C_D(q_i)}{\sum_{q_j\in D}C_D(q_j)}=\frac{C_D(q_i)}{|D|}$$

Counts of term q_i in the doc D

Length of the doc D

    - Bigram

$$P(q_i|q_j,D)=\frac{C_D(q_j,q_i)}{C_D(q_j)}$$

Counts of term pair (q_j,q_i) in the doc D

Counts of term q_j in the doc D

  - Similar formulas for corpus N-grams

$$P(q_i|Corpus)=\frac{C_{Corpus}(q_i)}{|Corpus|} \qquad P(q_i|q_j,D)=\frac{C_{Corpus}(q_j,q_i)}{C_{Corpus}(q_j)}$$

Corpus: an outside corpus or just the doc collection

HMM/N-gram-based Model

- Basically, m_1, m_2, m_3, m_4, can be estimated by using the Expectation-Maximization (EM) algorithm

because of the insufficiency of training data

  - All docs share the same weights here
  - The *N*-gram probability distributions also can be estimated using the EM algorithm instead of the maximum likelihood estimation
- For those docs with training queries, m_1, m_2, m_3, m_4, can be estimated by using the Minimum Classification Error (MCE) training algorithm
  - The docs can have different weights

HMM/N-gram-based Model

- Expectation-Maximum Training
  - The weights are tied among the documents
  - E.g. ***m_1* of Type I HMM** can be trained using the following equation:

the old weight

819 queries ≦2265 docs

the new weight

$$m_1 = \frac{\sum_{Q \in [TrainSet]_Q} \sum_{D \in [Doc]_{R\text{ to }Q}} \sum_{q_n \in Q} \left[\frac{\hat{m}_1 P(q_n|D)}{\hat{m}_1 P(q_n|D) + \hat{m}_2 P(q_n|Corpus)} \right]}{\sum_{Q \in [TrainSet]_Q} |Q| \cdot \left| [Doc]_{R\text{ to }Q} \right|}$$

- Where $[TrainSet]_Q$ is the set of training query exemplars, $[Doc]_{R\text{ to }Q}$ is the set of docs that are relevant to a specific training query exemplar Q, $|Q|$ is the length of the query, and $\left|[Doc]_{R\text{ to }Q}\right|$ is the total number of docs relevant to the query Q

HMM/N-gram-based Model

- Expectation-Maximum Training

Empirical Derivation

$$P(Q \mid D) > P(Q \mid \hat{D}) \ ?$$

The new model

The old model

$$\log P(Q \mid D) - \log P(Q \mid \hat{D})$$

$$= \sum_{q_n \in Q} \sum_{k} P(k \mid q_n, \hat{D}) \log \left[P(q_n \mid D) \frac{P(q_n, k \mid D)}{P(q_n, k \mid D)} \right] - \sum_{q_n \in Q} \sum_{k} P(k \mid q_n, \hat{D}) \log \left[P(q_n \mid \hat{D}) \frac{P(q_n, k \mid \hat{D})}{P(q_n, k \mid \hat{D})} \right]$$

$$= \sum_{q_n \in Q} \sum_{k} P(k \mid q_n, \hat{D}) \log \frac{P(q_n, k \mid D)}{P(k \mid q_n, D)} - \sum_{q_n \in Q} \sum_{k} P(k \mid q_n, \hat{D}) \log \frac{P(q_n, k \mid \hat{D})}{P(k \mid q_n, \hat{D})}$$

$$\Phi(D,\hat{D}) - \Phi(\hat{D},\hat{D})$$

$$= \sum_{q_n \in Q} \left[\sum_{k} P(k \mid q_n, \hat{D}) \log P(q_n, k \mid D) - \sum_{k} P(k \mid q_n, \hat{D}) \log P(q_n, k \mid \hat{D}) \right]$$

$$+ \sum_{q_n \in Q} \left[\sum_{k} P(k \mid q_n, \hat{D}) \log P(k \mid q_n, \hat{D}) - \sum_{k} P(k \mid q_n, \hat{D}) \log P(k \mid q_n, D) \right]$$

$$H(\hat{D},\hat{D}) - H(D,\hat{D}) \geq 0$$

$$\geq \sum_{q_n \in Q} \left[\sum_{k} P(k \mid q_n, \hat{D}) \log P(q_n, k \mid D) - \sum_{k} P(k \mid q_n, \hat{D}) \log P(q_n, k \mid \hat{D}) \right]$$

$$\left\{ \because \sum_i p_i \log q_i - \sum_i p_i \log p_i = \sum_i p_i \log \frac{q_i}{p_i} \leq \sum_i p_i \left(\frac{q_i}{p_i} - 1 \right) = 0 \right\}$$

Jensen's inequality

$$(\because \log x \leq x - 1)$$

$$\therefore \text{ If } \sum_{q_n \in Q} \sum_{k} P(k \mid q_n, \hat{D}) \log P(q_n, k \mid D) \geq \sum_{q_n \in Q} \sum_{k} P(k \mid q_n, \hat{D}) \log P(q_n, k \mid \hat{D})$$

$$\text{then } \ P(Q \mid D) > P(Q \mid \hat{D})$$

HMM/N-gram-based Model

- Expectation-Maximum Training

empirical distribution the model

Q function

$$\boxed{\Phi(D,\hat{D}) = \sum_{q_n \in Q}\sum_k P\left(k|q_n,\hat{D}\right)\log P\left(q_n,k|D\right)}$$

$$= \sum_{q_n \in Q}\sum_k \frac{P\left(q_n|k,\hat{D}\right)P\left(k|\hat{D}\right)}{P\left(q_n|\hat{D}\right)}\log\left[P\left(q_n|k,D\right)P\left(k|D\right)\right]$$

$$= \sum_{q_n \in Q}\sum_k \frac{P\left(q_n|k,\hat{D}\right)\hat{m}_k}{\sum_j P\left(q_n|j,\hat{D}\right)\hat{m}_j}\log\left[P\left(q_n|k,D\right)m_k\right]$$

$$\Phi'(D,\hat{D}) = \sum_{q_n \in Q}\sum_k \left\{\frac{P\left(q_n|k,\hat{D}\right)\hat{m}_k}{\sum_j P\left(q_n|j,\hat{D}\right)\hat{m}_j}\log\left[P\left(q_n|k,D\right)m_k\right]\right\} + \boxed{l\left(\sum_i m_i - 1\right)}$$

normalization constraints using Lagrange multipliers

$$\frac{\partial \Phi'(D,\hat{D})}{\partial m_k} = \frac{1}{m_k}\left[\sum_{q_n \in Q}\frac{P\left(q_n|k,\hat{D}\right)\hat{m}_k}{\sum_j P\left(q_n|j,\hat{D}\right)\hat{m}_j}\right] + l = 0$$

Assume $G_k = \sum_{q_n \in Q}\frac{P\left(q_n|k,\hat{D}\right)\hat{m}_k}{\sum_j P\left(q_n|j,\hat{D}\right)\hat{m}_j} \Rightarrow \frac{G_1}{m_1} = \frac{G_2}{m_2} = ... = \frac{G_k}{m_k} = ... = -l$

$$\therefore\ l = -\sum_s G_s \qquad \therefore\ m_k = \frac{\overbrace{\boxed{\sum_{q_n \in Q}\frac{P\left(q_n|k,\hat{D}\right)\hat{m}_k}{\sum_j P\left(q_n|j,\hat{D}\right)\hat{m}_j}}}^{G_k}}{\underbrace{\boxed{\sum_s\sum_{q_n \in Q}\frac{P\left(q_n|k,\hat{D}\right)\hat{m}_s}{\sum_j P\left(q_n|j,\hat{D}\right)\hat{m}_j}}}_{l}} = \frac{\sum_{q_n \in Q}\frac{P\left(q_n|k,\hat{D}\right)\hat{m}_k}{\sum_j P\left(q_n|j,\hat{D}\right)\hat{m}_j}}{|Q|}$$

HMM/N-gram-based Model

- Experimental results with EM training
  - HMM/N-gram-based approach

Average Precision		Word-level			Syllable-level		
		Uni	Uni+Bi	Uni+Bi*	Uni	Uni+Bi	Uni+Bi*
TDT2	TQ/TD	**0.6327**	0.6069	0.5427	0.4698	0.5220	0.5718
	TQ/SD	0.5658	**0.5702**	0.4803	0.4411	0.5011	0.5307
TDT3	TQ/TD	**0.6569**	0.6542	0.6141	0.5343	0.5970	0.6560
	TQ/SD	0.6308	0.6361	0.5808	0.5177	0.5678	**0.6433**

  - Vector space model

Average Precision		Word-level		Syllable-level	
		S(N), *N*=1	*S(N)*, *N*=1~2	*S(N)*, *N*=1	*S(N)*, *N*=1~2
TDT2	TQ/TD	0.5548	**0.5623**	0.3412	0.5254
	TQ/SD	0.5122	**0.5225**	0.3306	0.5077
TDT3	TQ/TD	0.6505	**0.6531**	0.3963	0.6502
	TQ/SD	0.6216	0.6233	0.3708	**0.6353**

  - HMM/N-gram-based approach is consistently better than vector space model

Review: The EM Algorithm

- Introduction of EM (Expectation Maximization):
  - Why EM?
    - Simple optimization algorithms for likelihood function relies on the intermediate variables, called latent (隱藏的)data
      In our case here, ***the state sequence** is the latent data*
    - Direct access to the data necessary to estimate the parameters is impossible or difficult
  - Two Major Steps :
    - ***E*** : expectation with respect to the latent data using the current estimate of the parameters and conditioned on the observations
    - ***M***: provides a new estimation of the parameters according to ML (or MAP)

Review: The EM Algorithm

- The EM Algorithm is important to HMMs and other learning techniques
  - Discover new model parameters to maximize the log-likelihood of incomplete data $\log P(o|\lambda)$ by iteratively maximizing the expectation of log-likelihood from complete data $\log P(o,s|\lambda)$

- Example
  - The observable training data o
    - We want to maximize $P(o|\lambda)$, λ is a parameter vector
  - The hidden (unobservable) data s
    - E.g. the component densities of observable data o, or the underlying state sequence in HMMs

$$\Theta(\lambda, \overline{\lambda}) = \sum_{o} E_{s|o,\lambda}\left[\log P(o,s|\lambda)\right]$$

HMM/N-gram-based Model

- Minimum Classification Error (MCE) Training
  - Given a query Q and a desired relevant doc D^*, define **the classification error function** as:

$$E(Q, D^*) = \frac{1}{|Q|}\left[-\log P\left(Q \middle| D^* \text{ is } R\right) + \max_{D'} \log P\left(Q \middle| D' \text{ is not } R\right)\right]$$

    - >0 means misclassified; <=0 means a correct decision
  - Transform the error function to the loss function

$$L(Q, D^*) = \frac{1}{1 + \exp(-\alpha E(Q, D^*) + \beta)}$$

    - In the range between 0 and 1

HMM/N-gram-based Model

- Minimum Classification Error (MCE) Training
  - Apply the loss function to the MCE procedure for iteratively updating the weighting parameters
    - Constraints:

$$m_k \geq 0\,, \quad \sum_k m_k = 1$$

    - Parameter Transforms, (e.g.,Type I HMM)

$$m_1 = \frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} \quad \text{and} \quad m_2 = \frac{e^{\tilde{m}_2}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}}$$

  - Iteratively update m_1 (e.g., Type I HMM)

$$\tilde{m}_1(i+1) = \tilde{m}_1(i) - \boxed{\varepsilon(i) \cdot \frac{\partial L(Q, D^*)}{\partial \tilde{m}_1}}\Bigg|_{D^* = D^*(i)}$$

    - Where,

$$\nabla_{D^*, \tilde{m}_1} = \varepsilon(i) \cdot \frac{\partial L(Q, D^*)}{\partial \tilde{m}_1}$$

$$= \varepsilon(i) \cdot \frac{\partial L(Q, D^*)}{\partial E(Q, D^*)} \cdot \frac{\partial E(Q, D^*)}{\partial \tilde{m}_1}, \quad \frac{\partial L(Q, D^*)}{\partial E(Q, D^*)} = \alpha \cdot L(Q, D^*) \cdot \left[1 - L(Q, D^*)\right]$$

HMM/N-gram-based Model

- Minimum Classification Error (MCE) Training
  - Iteratively update m_1 (e.g., Type I HMM)

$$\frac{\partial E(Q,D^*)}{\partial \tilde{m}_1} = \frac{-1}{|Q|} \frac{\partial \left\{ \sum_{q_n \in Q} \log \left[\frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P\left(q_n \middle| D^*\right) + \frac{e^{\tilde{m}_2}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P\left(q_n \middle| Corpus\right) \right] \right\}}{\partial \tilde{m}_1}$$

$$= \frac{-1}{|Q|} \sum_{q_n \in Q} \left\{ \frac{ \frac{-e^{\tilde{m}_1}}{\left(e^{\tilde{m}_1} + e^{\tilde{m}_2}\right)^2} \left[e^{\tilde{m}_1} P\left(q_n \middle| D^*\right) + e^{\tilde{m}_2} P\left(q_n \middle| Corpus\right) \right] + \frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P\left(q_n \middle| D^*\right) }{ \frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P\left(q_n \middle| D^*\right) + \frac{e^{\tilde{m}_2}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P\left(q_n \middle| Corpus\right) } \right\}$$

$$= \frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} - \frac{1}{|Q|} \sum_{q_n \in Q} \left\{ \frac{ \frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P\left(q_n \middle| D^*\right) }{ \frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P\left(q_n \middle| D^*\right) + \frac{e^{\tilde{m}_2}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P\left(q_n \middle| Corpus\right) } \right\}$$

$$= -\left[-m_1 + \frac{1}{|Q|} \sum_{q_n \in Q} \frac{m_1 P\left(q_n \middle| D^*\right)}{m_1 P\left(q_n \middle| D^*\right) + m_2 P\left(q_n \middle| Corpus\right)} \right],$$

HMM/N-gram-based Model

- Minimum Classification Error (MCE) Training
  - Iteratively update m_1 (e.g., Type I HMM)

$$\nabla_{D^*,\tilde{m}_1}(i) = -\varepsilon(i)\cdot\alpha\cdot L(Q,D^*)\cdot\left[1-L(Q,D^*)\right]$$
$$\cdot\left[-m_1(i)+\frac{1}{|Q|}\sum_{q_n\in Q}\frac{m_1(i)P(q_n|D^*)}{m_1(i)P(q_n|D^*)+m_2(i)P(q_n|Corpus)}\right],$$

the new weight

$$m_1(i+1)=\frac{e^{\tilde{m}_1(i+1)}}{e^{\tilde{m}_1(i+1)}+e^{\tilde{m}_2(i+1)}}$$

$$\tilde{m}_1(i+1)=\tilde{m}_1(i)-\nabla_{D^*,\tilde{m}_1}(i)$$

$$=\frac{e^{\tilde{m}_1(i)}e^{-\nabla_{D^*,\tilde{m}_1}(i)}}{e^{\tilde{m}_1(i)}e^{-\nabla_{D^*,\tilde{m}_1}(i)}+e^{\tilde{m}_2(i)}e^{-\nabla_{D^*,\tilde{m}_2}(i)}}$$

$$=\frac{e^{\tilde{m}_1(i)}e^{-\nabla_{D^*,\tilde{m}_1}(i)}\Big/\left(e^{\tilde{m}_1(i)}+e^{\tilde{m}_2(i)}\right)}{\left[e^{\tilde{m}_1(i)}e^{-\nabla_{D^*,\tilde{m}_1}(i)}\Big/\left(e^{\tilde{m}_1(i)}+e^{\tilde{m}_2(i)}\right)\right]+\left[e^{\tilde{m}_2(i)}e^{-\nabla_{D^*,\tilde{m}_2}(i)}\Big/\left(e^{\tilde{m}_1(i)}+e^{\tilde{m}_2(i)}\right)\right]}$$

the old weight

$$=\frac{m_1(i)\cdot e^{-\nabla_{D^*,\tilde{m}_1}(i)}}{m_1(i)\cdot e^{-\nabla_{D^*,\tilde{m}_1}(i)}+m_2(i)\cdot e^{-\nabla_{D^*,\tilde{m}_2}(i)}},$$

HMM/N-gram-based Model

- Minimum Classification Error (MCE) Training
  - Final Equations
    - Iteratively update m_1

$$\nabla_{D^*,\tilde{m}_1}(i) = -\varepsilon(i)\cdot\alpha\cdot L(Q,D^*)\cdot\left[1-L(Q,D^*)\right]$$
$$\cdot\left[-m_1(i)+\frac{1}{|Q|}\sum_{q_n\in Q}\frac{m_1(i)P(q_n|D^*)}{m_1(i)P(q_n|D^*)+m_2(i)P(q_n|Corpus)}\right]$$

$$m_1(i+1)=\frac{m_1(i)\cdot e^{-\nabla_{D^*,\tilde{m}_1}(i)}}{m_1(i)\cdot e^{-\nabla_{D^*,\tilde{m}_1}(i)}+m_2(i)\cdot e^{-\nabla_{D^*,\tilde{m}_2}(i)}}$$

    - m_2 can be updated in the similar way

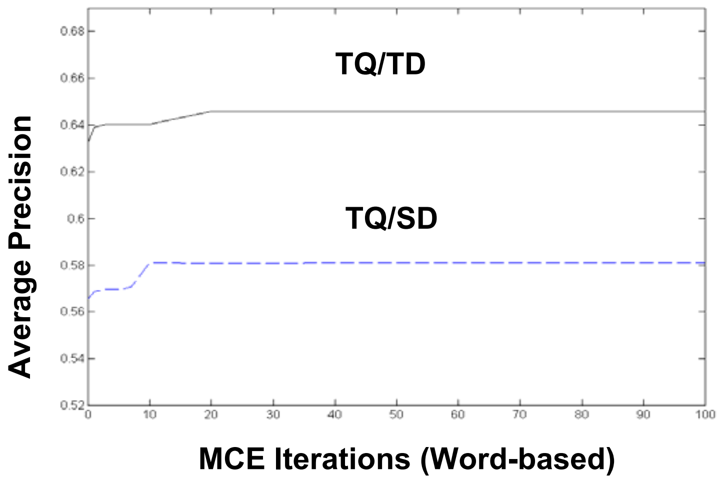
HMM/N-gram-based Model

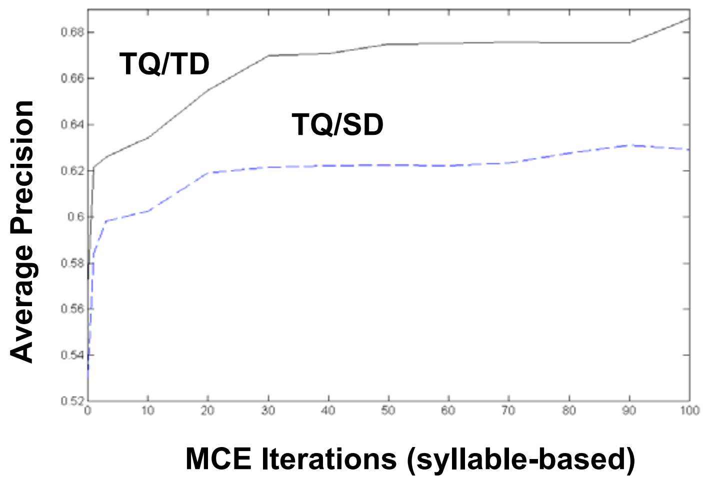
- Experimental results with MCE training

Before MCE Training

Average Precision		Word-level	Syllable-level	Fusion
		Uni	Uni+Bi*	
TDT2	TQ/TD	0.6459 (0.6327)	0.6858 (0.5718)	0.7329
	TQ/SD	0.5810 (0.5658)	0.6300 (0.5307)	0.6914

Iterations=100





  - The results for the syllable-level index features were significantly improved

HMM/N-gram-based Model

- Advantages
  - A formal mathematic framework
  - Use collection statistics but not heuristics
  - The retrieval system can be gradually improved through usage

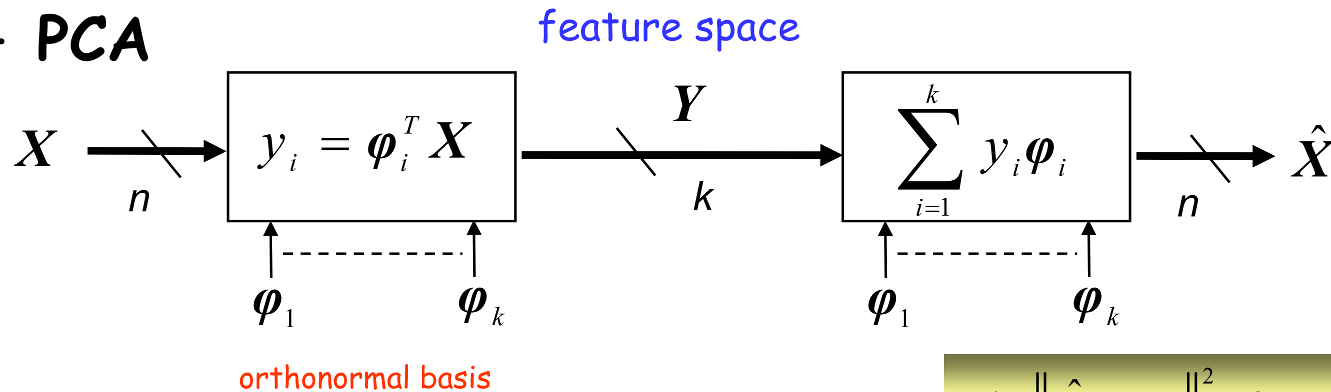
- Disadvantages
  - Only literal term matching (or word overlap measure)
    - The issue of *relevance* or *aboutness* is not taken into consideration
  - The implementation relevance feedback or query expansion is not straightforward

Latent Semantic Indexing (LSI)

- LSI: a technique projects queries and docs into a space with "latent" semantic dimensions
  - Co-occurring terms are projected onto the same dimensions
  - In the latent semantic space (with fewer dimensions), a query and doc can have high cosine similarity even if they do not share any terms
  - Dimensions of the reduced space correspond to the axes of greatest variation
    - Closely related to *Principal Component Analysis* (PCA)

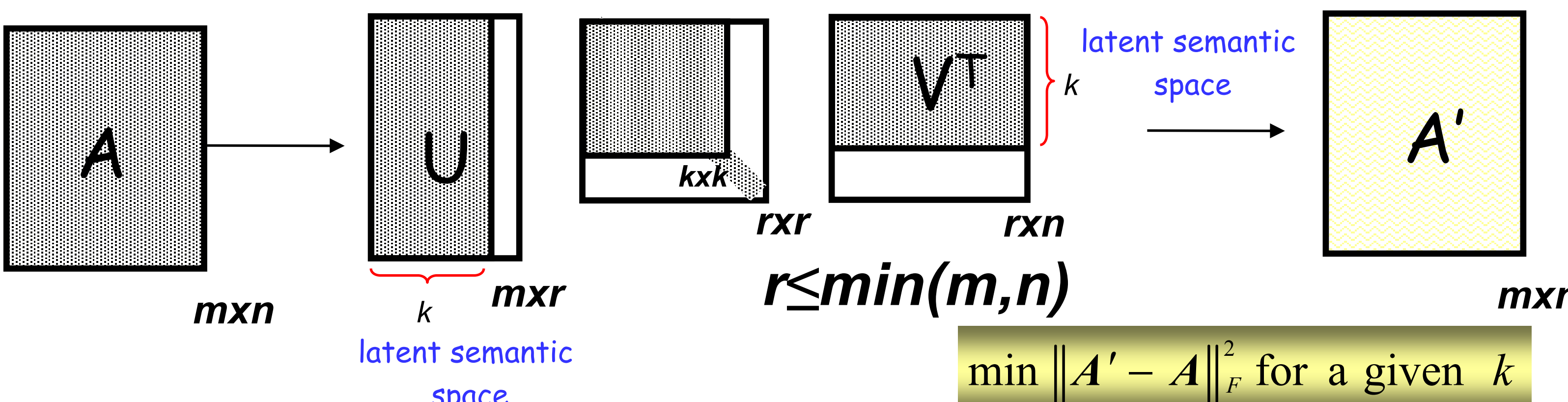
Latent Semantic Indexing (LSI)

- Dimension Reduction and Feature Extraction
  - PCA



$$\min \left\| \hat{X} - X \right\|^2 \quad \text{for a given } k$$

  - SVD (in LSI)



$$\min \left\| A' - A \right\|_F^2 \quad \text{for a given } k$$

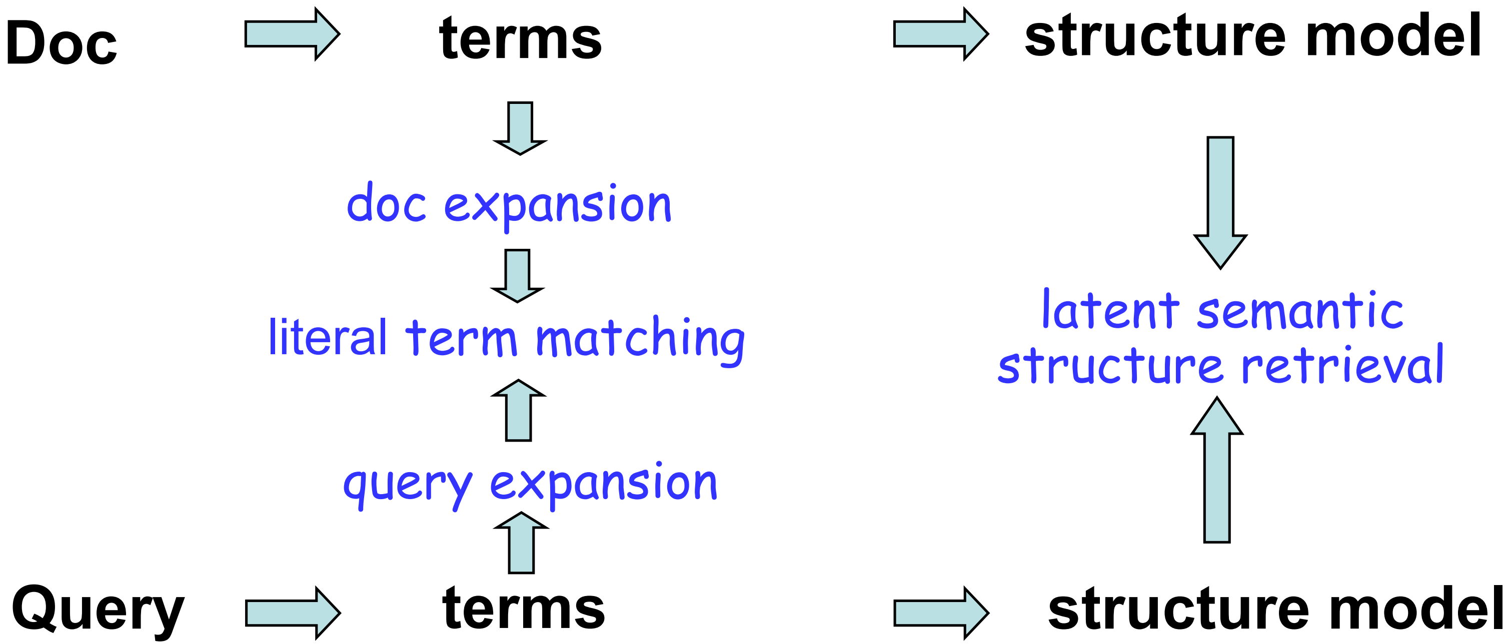
Latent Semantic Indexing (LSI)

- Singular Value Decomposition (SVD) used for the word-document matrix
  - *A least-squares method for dimension reduction*

	Term 1	Term 2	Term 3	Term 4
Query	user	interface		
Document 1	user	interface	HCI	interaction
Document 2			HCI	interaction

Latent Semantic Indexing (LSI)

- Frameworks to circumvent vocabulary mismatch



Latent Semantic Indexing (LSI)

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

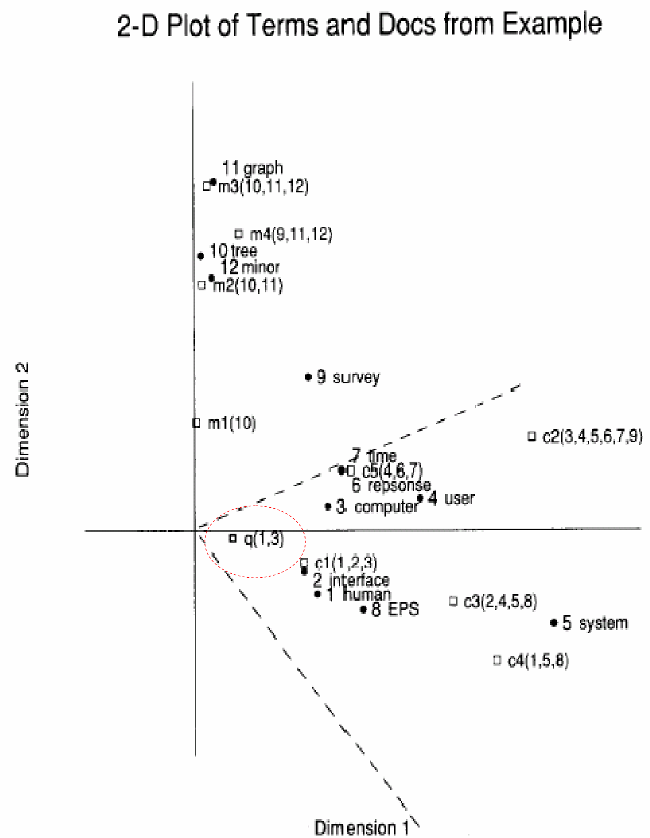
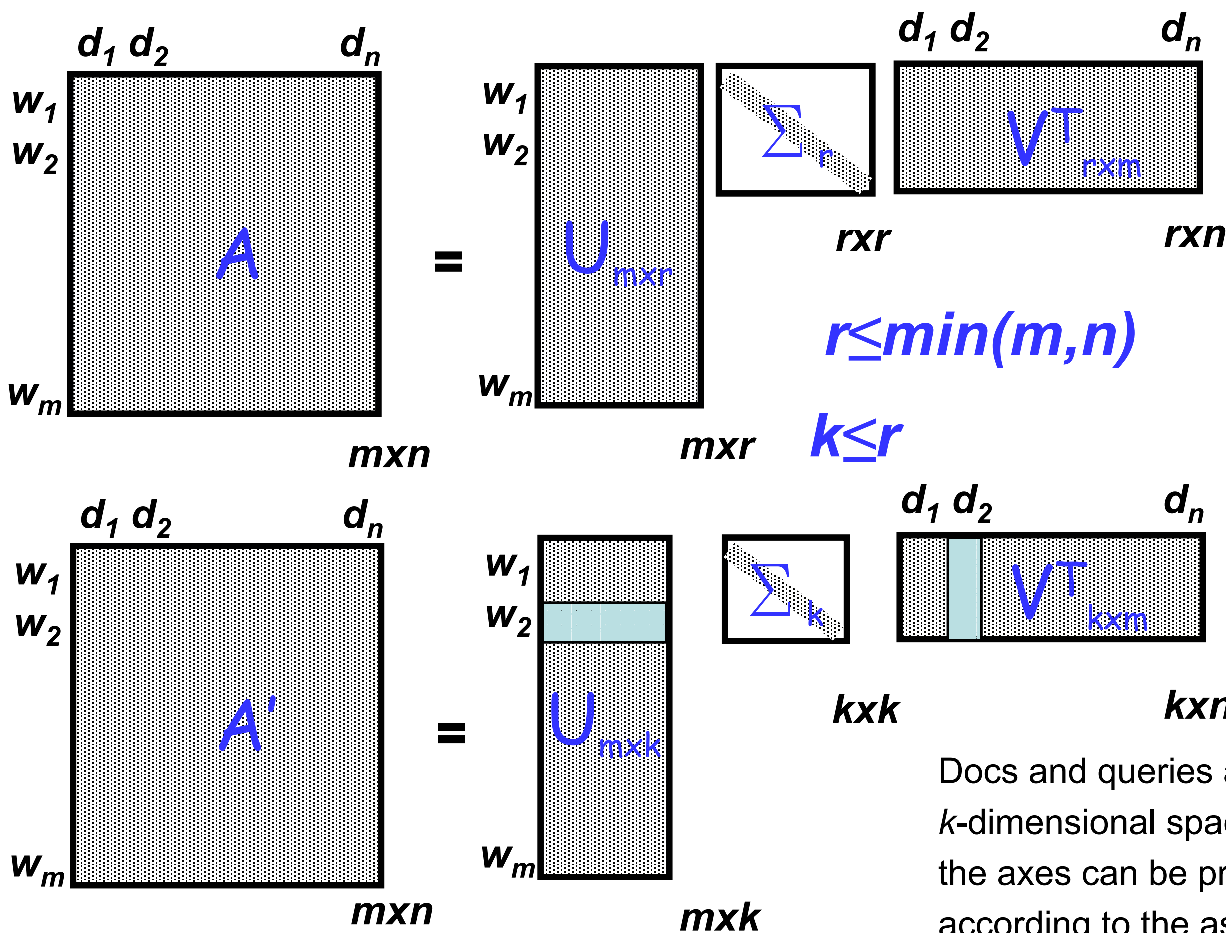


FIG. 1. A two-dimensional plot of 12 Terms and 9 Documents from the sampe TM set. Terms are represented by filled circles. Documents are shown as open squares, and component terms are indicated parenthetically. The query ("human computer interaction") is represented as a pseudo-document at point *q*. Axes are scaled for Document-Document or Term-Term comparisons. The dotted cone represents the region whose points are within a cosine of .9 from the query *q*. All documents about human-computer (c1–c5) are "near" the query (i.e., within this cone), but none of the graph theory documents (m1–m4) are nearby. In this reduced space, even documents c3 and c5 which share no terms with the query are near it.

Latent Semantic Indexing (LSI)

- Singular Value Decomposition (SVD)



Both U and V has orthonormal column vectors

$U^T U = I_{kXk}$

$V^T V = I_{kXk}$

$r \leq min(m,n)$

$k \leq r$

$||A||_F{}^2 \geq ||A'||\ _F{}^2$

Docs and queries are represented in a *k*-dimensional space. The quantities of the axes can be properly weighted according to the associated diagonal values of Σ_k

Latent Semantic Indexing (LSI)

- Singular Value Decomposition (SVD)
  - $A^T A$ is symmetric $n \times n$ matrix
    - All eigenvalues λ_j are nonnegative real numbers

$$\lambda_1 \geq \lambda_2 \geq .... \geq \lambda_n \geq 0 \qquad \Sigma^2 = diag(\lambda_1, \lambda_1, ..., \lambda_n)$$

    - All eigenvectors v_j are orthonormal

$$V = [v_1 v_2 ... v_n] \qquad v_j^T v_j = 1 \qquad (V^T V = I_{nxn})$$

    - Define **singular values** $\sigma_j = \sqrt{\lambda_j},\ j = 1, ..., n$
      - As the square roots of the eigenvalues of $A^T A$
      - As the lengths of the vectors Av_1, Av_2, …., Av_n

For $\lambda_i \neq 0$, i=1,…r,
$\{Av_1, Av_2, …., Av_r\}$ is an orthogonal basis of Col A

$$\sigma_1 = \|Av_1\|$$

$$\sigma_2 = \|Av_2\|$$

.....

Latent Semantic Indexing (LSI)

- $\{Av_1, Av_2, \dots, Av_r\}$ is an orthogonal basis of Col A

$$Av_i \bullet Av_j = (Av_i)^T Av_j = v_i^T A^T Av_j = \lambda_j v_i^T v_j = 0$$

  – Suppose that A (or $A^T A$) has rank $r \leq n$

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_r > 0, \quad \lambda_{r+1} = \lambda_{r+2} = \dots = \lambda_r = 0$$

  – Define an orthonormal basis $\{u_1, u_2, \dots, u_r\}$ for Col A

$$u_i = \frac{1}{\|Av_i\|} Av_i = \frac{1}{\sigma_i} Av_i \Rightarrow \sigma_i u_i = Av_i$$

$$\Rightarrow [u_1\ u_2 \dots u_r]\Sigma_r = A[v_1\ v_2 \quad v_r]$$

    - Extend to an orthonormal basis $\{u_1, u_2, \dots, u_m\}$ of R^m

$$\Rightarrow [u_1\ u_2 \dots u_r \dots u_m]\Sigma = A[v_1\ v_2 \dots v_r \dots v_m]$$

$$\Rightarrow U\Sigma = AV$$

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

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

$$\|A\|_F^2 = \sigma_1^2 + \sigma_2^2 + \dots + \sigma_r^2 \quad ?$$

Latent Semantic Indexing (LSI)

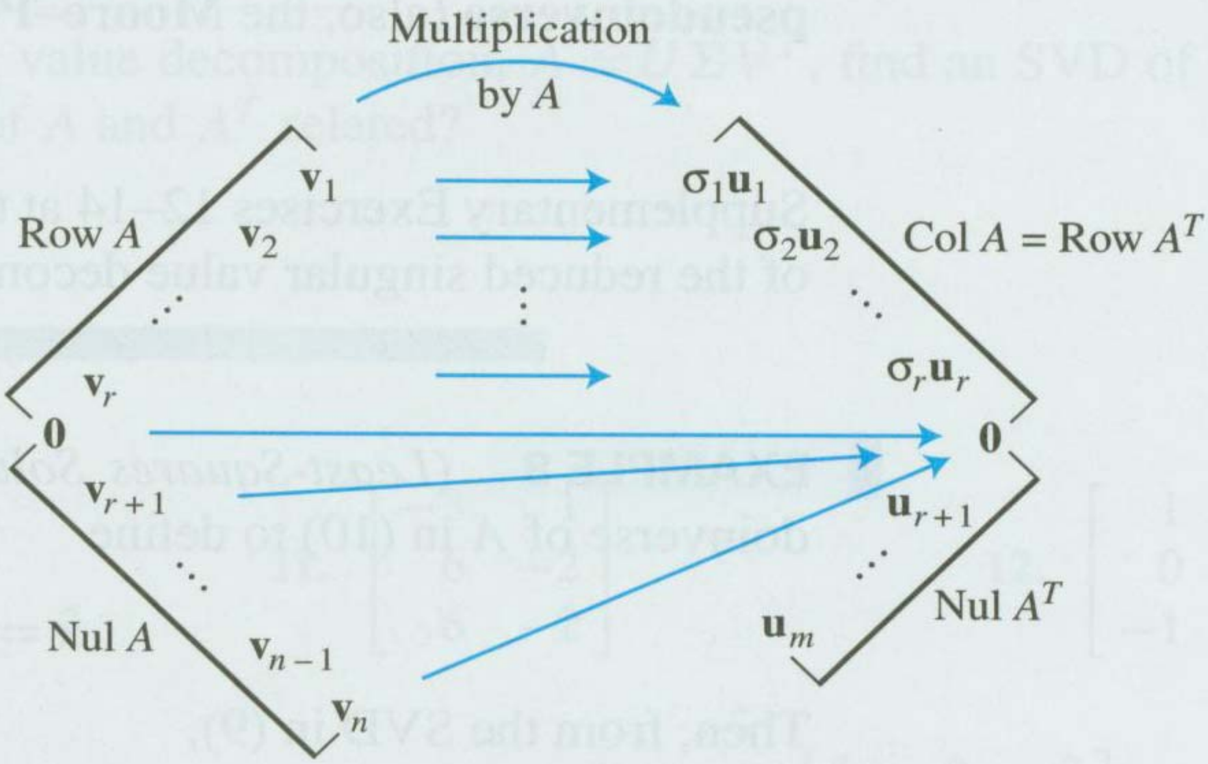
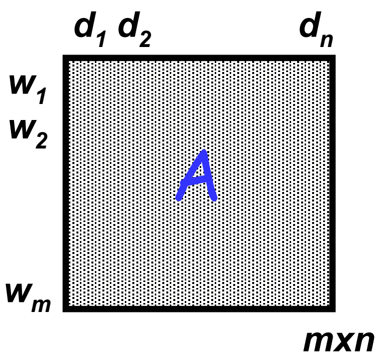


FIGURE 4 The four fundamental subspaces and the action of A.

Latent Semantic Indexing (LSI)

- Fundamental comparisons based on SVD
  - The original word-document matrix (A)



- compare two terms $\rightarrow$ dot product of two rows of A
  - or an entry in AA^T
- compare two docs $\rightarrow$ dot product of two columns of A
  - or an entry in A^TA
- compare a term and a doc $\rightarrow$ each individual entry of A

  - The new word-document matrix (A')

$U'=U_{m\times k}$
$\Sigma'=\Sigma_k$
$V'=V_{n\times k}$

- compare two terms $A'A'^T=(U'\Sigma'V'^T)(U'\Sigma'V'^T)^T=U'\Sigma'\underbrace{V'^TV'}_{I}\Sigma'^TU'^T=(U'\Sigma')(U'\Sigma')^T$
  $\rightarrow$ dot product of two rows of $U'\Sigma'$

For stretching or shrinking

- compare two docs $A'^TA'=(U'\Sigma'V'^T)^T{}'(U'\Sigma'V'^T)=V'\Sigma'^T\overbrace{U^TU'}^{I}\Sigma'V'^T=(V'\Sigma')(V'\Sigma')^T$
  $\rightarrow$ dot product of two rows of $V'\Sigma'$
- compare a query and a doc $\rightarrow$ each individual entry of A'

Latent Semantic Indexing (LSI)

- **Fold-in**: find representations for pesudo-docs *q*
  - For objects (new queries or docs) that did not appear in the original analysis
    - Fold-in a new *m*x1 query (or doc) vector

$$\hat{q}_{1xk} = \boxed{\left(q^T\right)_{1xm} U_{m\times k}} \boxed{\Sigma^{-1}_{k\times k}}$$

Just like a row of V

Query represented by the weighted sum of it constituent term vectors

The separate dimensions are differentially weighted

  - Cosine measure between the query and doc vectors in the latent semantic space

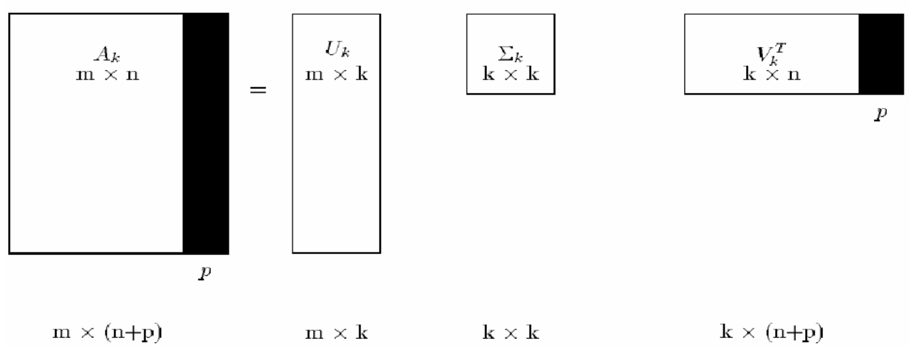
$$sim\left(\hat{q},\hat{d}\right) = coine\ (\hat{q}\Sigma,\hat{d}\Sigma) = \frac{\hat{q}\Sigma^2\hat{d}^T}{|\hat{q}\Sigma|\left|\hat{d}\Sigma\right|}$$

row vectors

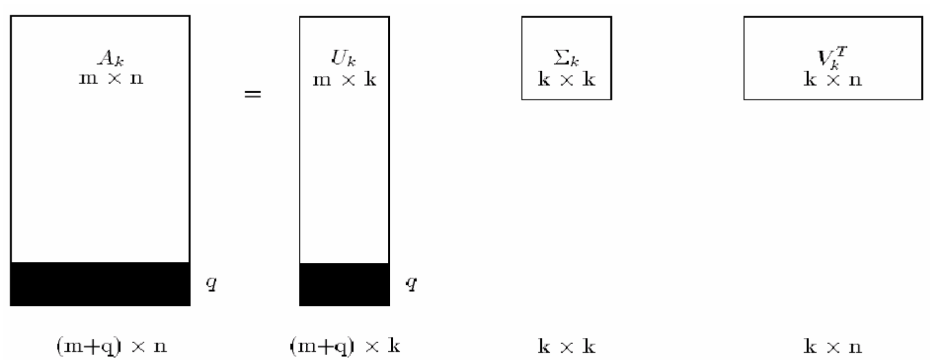
Latent Semantic Indexing (LSI)

- Fold-in a new 1xn term vector

$$\hat{t}_{1\,xk} = t_{1\,xn} V_{n \times k} \Sigma^{-1}_{k \times k}$$



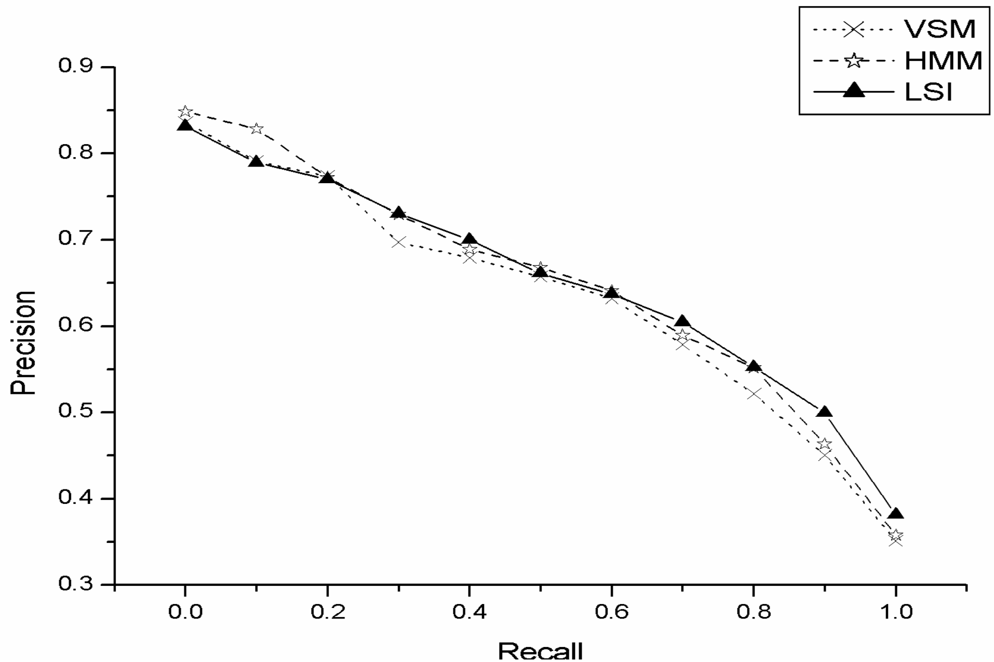
Mathematical representation of folding-in p documents.



Mathematical representation of folding-in q terms.

Latent Semantic Indexing (LSI)

- Experimental results
  - HMM is consistently better than VSM at all recall levels
  - LSI is better than VSM at higher recall levels



Recall-Precision curve at 11 standard recall levels evaluated on TDT-3 SD collection. (Using word-level indexing terms)

Latent Semantic Indexing (LSI)

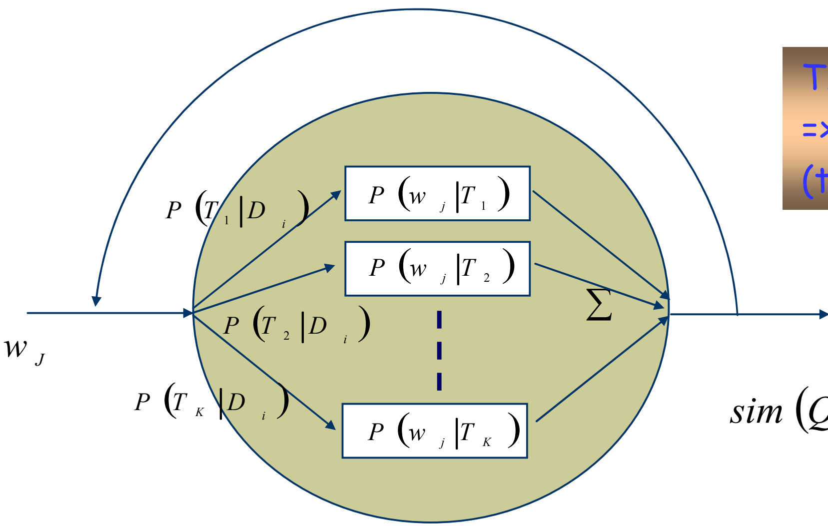
- Advantages
  - A clean formal framework and a clearly defined optimization criterion (least-squares)
    - Conceptual simplicity and clarity
  - Handle synonymy problems ("heterogeneous vocabulary")
  - Good results for high-recall search
    - Take term co-occurrence into account
- Disadvantages
  - High computational complexity
  - LSI offers only a partial solution to polysemy
    - E.g. bank, bass,…

Probabilistic Latent Semantic Analysis (PLSA)

Thomas Hofmann 1999

- Also called The Aspect Model, Probabilistic Latent Semantic Indexing (PLSA)
  - Can be viewed as a complex HMM Model

The latent variables
=>The unobservable class variables T_i
(topics or domains)



$$Q = w_1 w_2 \ldots w_j \ldots w_J$$

$$sim(Q,D_i) = P(Q|D_i) = \prod_{w_j} P(w_j|D_i)$$

$$= \prod_{w_j}\left[\sum_{k=1}^{K} P(w_j, T_k|D_i)\right]$$

?

$$= \prod_{w_j}\left[\sum_{k=1}^{K} P(w_j|T_k)P(T_k|D_i)\right]$$

$$sim(Q,D_i) = P(D_i|Q) = \frac{P(Q,D_i)}{P(Q)} \approx P(Q,D_i) = P(Q|D_i)\cancel{P(D_i)} \approx P(Q|D_i)$$

$$\Rightarrow sim(Q,D_i) \approx P(Q|D_i)$$

Probabilistic Latent Semantic Analysis (PLSA)

- Definition
  - $P(D_i)$: the prob. when selecting a doc D_i
  - $P(T_k|D_i)$: the prob. when pick a latent class T_k for the doc D_i
  - $P(w_j|T_k)$: the prob. when generating a word w_j from the class T_k

Probabilistic Latent Semantic Analysis (PLSA)

- Assumptions
  - **Bag-of-words**: treat docs as *memoryless* source, words are generated independently
  - **Conditional independent**: the doc D_i and word w_j are independent conditioned on the state of the associated latent variable T_k

$$P(w_j, D_i | T_k) \approx P(w_j | T_k) P(D_i | T_k)$$

$$\Rightarrow \quad P(w_j | D_i) = \sum_{k=1}^{K} P(w_j, T_k | D_i) = \sum_{k=1}^{K} \frac{P(w_j, D_i, T_k)}{P(D_i)} = \sum_{k=1}^{K} \frac{P(w_j, D_i | T_k) P(T_k)}{P(D_i)}$$

$$= \sum_{k=1}^{K} \frac{P(w_j | T_k) P(D_i | T_k) P(T_k)}{P(D_i)} = \sum_{k=1}^{K} \frac{P(w_j | T_k) P(T_k, D_i)}{P(D_i)}$$

$$= \sum_{k=1}^{K} P(w_j | T_k) P(T_k | D_i)$$

Probabilistic Latent Semantic Analysis (PLSA)

- Probability estimation using EM (expectation-maximization) algorithm
  - **E** (expectation) step

take expectation

complete data likelihood

$$E\left[L^C\right]=\sum_{D_i}\sum_{w_j} n(w_j,D_i)\, E_{T_k|w_j,D_i}\left[\log P(w_j,T_k|D_i)\right]$$

$$=\sum_{D_i}\sum_{w_j} n(w_j,D_i)\sum_{T_k}\left[\underbrace{\hat{P}(T_k|w_j,D_i)}_{\text{empirical distribution}}\log \underbrace{P(w_j,T_k|D_i)}_{\text{the model}}\right]$$

$$\left\{\hat{P}(T_k|w_j,D_i)=\frac{\hat{P}(T_k,w_j|D_i)}{\hat{P}(w_j|D_i)}=\frac{\hat{P}(w_j|T_k)\hat{P}(T_k|D_i)}{\sum_{T_k}\hat{P}(w_j|T_k)\hat{P}(T_k|D_i)}\right\}$$

$$=\sum_{D_i}\sum_{w_j} n(w_j,D_i)\sum_{T_k}\left[\hat{P}(T_k|w_j,D_i)\log P(w_j|T_k)P(T_k|D_i)\right]$$

Kullback-Leibler divergence

$$=\sum_{D_i}\sum_{w_j} n(w_j,D_i)\sum_{T_k}\left[\frac{\hat{P}(w_j|T_k)\hat{P}(T_k|D_i)}{\sum_{T_k}\hat{P}(w_j|T_k)\hat{P}(T_k|D_i)}\log P(w_j|T_k)P(T_k|D_i)\right]$$

Probabilistic Latent Semantic Analysis (PLSA)

- Probability estimation using EM
  - **M** (maximization) step

$$Q = E\left[L^C\right] + \boxed{\sum_{T_k} \tau_k \left(1 - \sum_{w_j} P\left(w_j \middle| T_k\right)\right) + \sum_{D_i} \rho_i \left(1 - \sum_{T_k} P\left(T_k \middle| D_i\right)\right)}$$

normalization constraints using Lagrange multipliers

$$Q_{P\left(w_j | T_k\right)} = \sum_{D_i} \sum_{w_j} n\left(w_j, D_i\right) \hat{P}\left(T_k \middle| w_j, D_i\right) \log P\left(w_j \middle| T_k\right) + \tau_k \left(1 - \sum_{w_j} P\left(w_j \middle| T_k\right)\right)$$

$$Q_{P\left(T_k | D_j\right)} = \sum_{w_j} n\left(w_j, D_i\right) \sum_{T_k} \hat{P}\left(T_k \middle| w_j, D_i\right) \log P\left(T_k \middle| D_i\right) + \rho_j \left(1 - \sum_{T_k} P\left(T_k \middle| D_i\right)\right)$$

Probabilistic Latent Semantic Analysis (PLSA)

- Probability estimation using EM
  - **M** (maximization) step
    - Take differentiation

The training formula

$$P(w_j|T_k) = \boxed{\frac{\sum_{D_i} n(w_j, D_i)\hat{P}(T_k|w_j, D_i)}{\sum_{w_j}\sum_{D_i} n(w_j, D_i)\hat{P}(T_k|w_j, D_i)}}$$

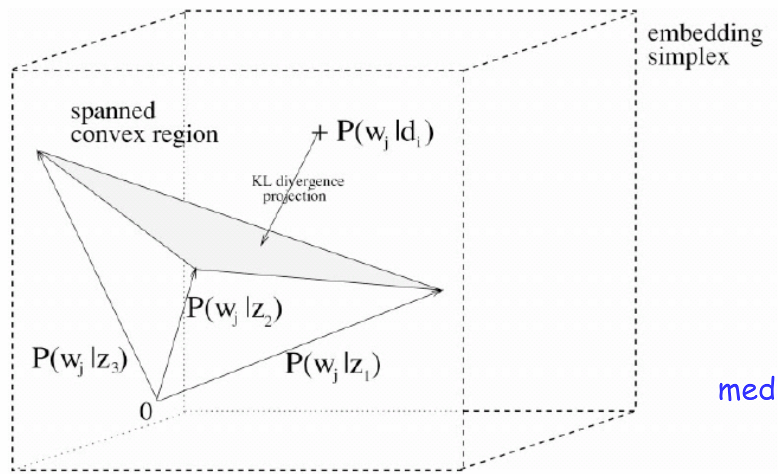
$$P(T_k|D_j) = \frac{\sum_{w_j} n(w_j, D_i)\hat{P}(T_k|w_j, D_i)}{\sum_{T_k}\sum_{w_j} n(w_j, D_i)\hat{P}(T_k|w_j, D_i)} = \frac{\sum_{w_j} n(w_j, D_i)\hat{P}(T_k|w_j, D_i)}{\sum_{w_j} n(w_j, D_i)}$$

$$= \boxed{\frac{\sum_{w_j} n(w_j, D_i)\hat{P}(T_k|w_j, D_i)}{n(D_i)}}$$

The training formula

Probabilistic Latent Semantic Analysis (PLSA)

- Latent Probability Spaces



Sketch of the probability simplex and a convex region spanned by class-conditional probabilities in the aspect model.

Dimensionality K=128 (latent classes)

Aspect 1	Aspect 2	Aspect 3	Aspect 4
imag	video	region	speaker
SEGMENT	sequenc	contour	speech
textur	motion	boundari	recogni
color	frame	descrip	signal
tissu	scene	imag	train
brain	SEGMENT	SEGMENT	hmm
slice	shot	precis	sourc
cluster	imag	estim	speakerindepend
mri	cluster	pixel	SEGMENT
algorithm	visual	paramet	sound

medical imaging | image sequence analysis | context of contour boundary detection | phonetic segmentation

$$P(w_j, D_i) = \sum_{T_k} P(w_j, T_k, D_i) = \sum_{T_k} P(w_j | T_k, D_i) P(T_k, D_i)$$

$$= \sum_{T_k} P(w_j | T_k) P(T_k) P(D_i | T_k)$$

$$\boldsymbol{P}(\boldsymbol{W}, \boldsymbol{D}) = \hat{U} : \left(P(w_j | T_k)\right)_{j,k} \cdot \hat{\Sigma} : \mathrm{diag}\left(P(T_k)\right)_k \cdot \hat{V} : \left(P(D_i | T_k)\right)_{i,k}$$

Probabilistic Latent Semantic Analysis (PLSA)

- One more example on TDT1 dataset

aviation	space missions	family love	Hollywood love
Aspect 1	Aspect 2	Aspect 3	Aspect 4
plane	space	home	film
airport	shuttle	family	movie
crash	mission	like	music
flight	astronauts	love	new
safety	launch	kids	best
aircraft	station	mother	hollywood
air	crew	life	love
passenger	nasa	happy	actor
board	satellite	friends	entertainment
airline	earth	cnn	star

The 2 aspects to most likely generate the word 'flight' (left) and 'love' (right), derived from a $K = 128$ aspect model of the TDT1 document collection. The displayed terms are the most probable words in the class-conditional distribution $P(w_j | z_k)$, from top to bottom in descending order.

Probabilistic Latent Semantic Analysis (PLSA)

- Comparison with LSI
  - Decomposition/Approximation
    - **LSI**: least-squares criterion measured on the L2- or Frobenius norms of the word-doc matrices
    - **PLSA**: maximization of the likelihoods functions based on the cross entropy or Kullback-Leibler divergence between the empirical distribution and the model
  - Computational complexity
    - LSI: SVD decomposition
    - PLSA: EM training, is time-consuming for iterations ?

Probabilistic Latent Semantic Analysis (PLSA)

- Experimental Results

PLSI-U*

– Two ways to smoothen empirical distribution with PLSI

- Combine the cosine score with that of the vector space model (so does LSI)
- Combine the multinomials individually

$$P_{PLSI-Q*}(\omega_j \mid d_i) = \lambda \boxed{P_{Empirical}(\omega_j \mid d_i)} + (1-\lambda) P_{PLSA}(\omega_j \mid d_i)$$

$$P_{Empirical}(\omega_j \mid d_i) = \frac{n(w_j, d_i)}{n(d_i)}$$

Both provide almost identical performance

- It's not known if PLSA was used alone

Probabilistic Latent Semantic Analysis (PLSA)

- Experimental Results

PLSI-Q*

– Use the low-dimensional representation $P(T_k \mid Q)$ and $P(T_k \mid D_i)$ (be viewed in a *k*-dimensional latent space) to evaluate relevance by means of cosine measure

– Combine the cosine score with that of the vector space model

– Use the ad hoc approach to reweight the different model components (dimensions) by

$$RW(T_k) = \sum_{w_j} \left[P(w_j \mid T_k) \cdot idf(w_j) \right]$$

$$sim(Q, D) = \frac{\sum_{w_j \in Q} \left[n(q, w_j) \sum_{T_k} RW^2(T_k) P(T_k \mid w_j) P(T_k \mid D_i) \right]}{\sqrt{\sum_{w_j \in Q} \left[n(q, w_j) \sum_{T_k} RW^2(T_k) P^2(T_k \mid w_j) \right]} \sqrt{\sum_{T_k} RW^2(T_k) P^2(T_k \mid D_i)}}$$

Probabilistic Latent Semantic Analysis (PLSA)

- Experimental Results