

# Statistical Modeling Approaches for Information Retrieval

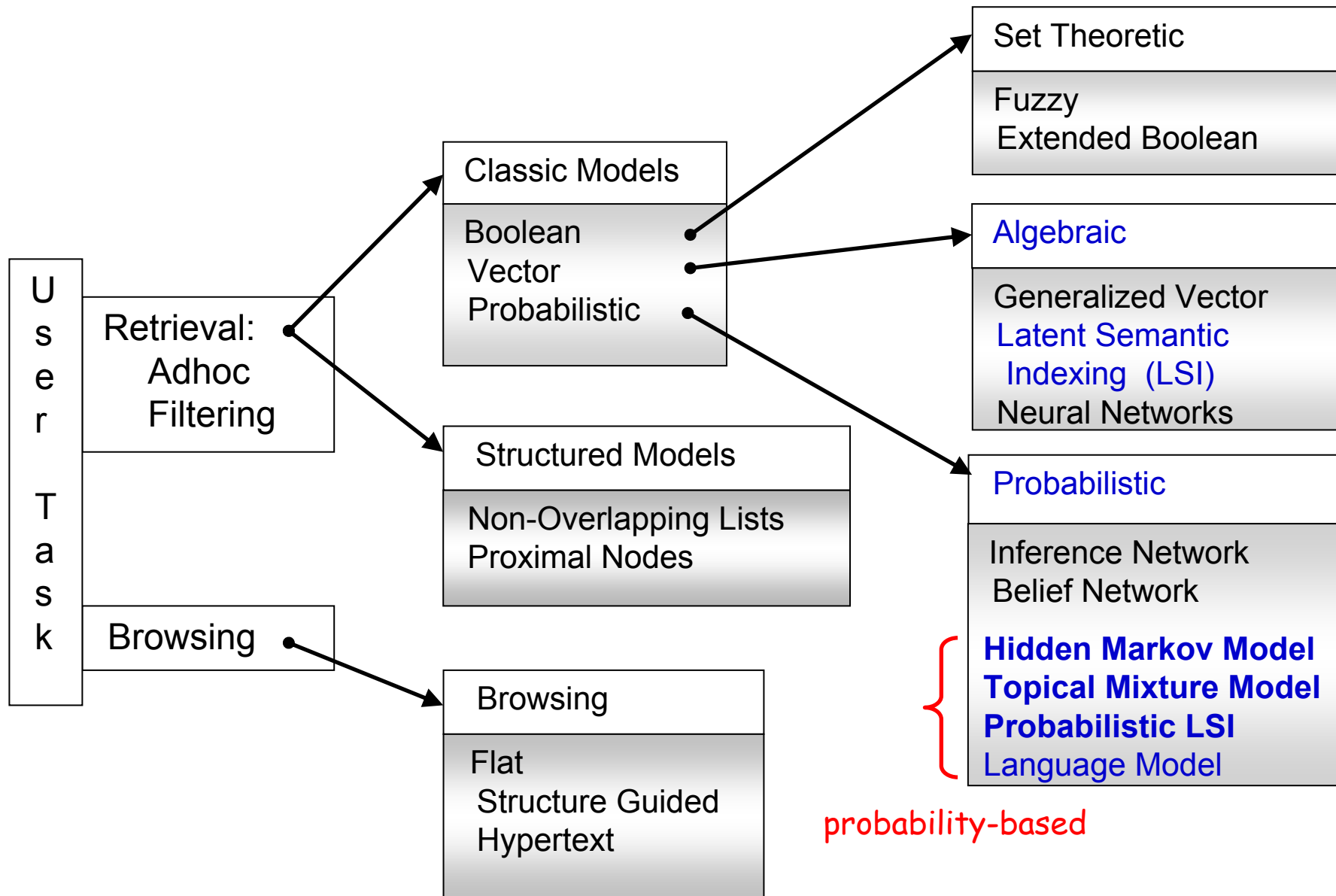
1. HMM/N-gram-based
2. Topical Mixture Model (TMM)
2. Latent Semantic Indexing (LSI)
3. Probabilistic Latent Semantic Analysis (PLSA)

Berlin Chen 2004

## References:

1. W. B. Croft and J. Lafferty (Editors). *Language Modeling for Information Retrieval*. July 2003
2. Berlin Chen et al. *A Discriminative HMM/N-Gram-Based Retrieval Approach for Mandarin Spoken Documents*. ACM Transactions on Asian Language Information Processing, June 2004
3. Berlin Chen. *Exploring the Use of Latent Topical Information for Statistical Chinese Spoken Document Retrieval*, 2004
4. M. W. Berry et al. *Using Linear Algebra for Intelligent Information Retrieval*. Technical report, 1994
5. Thomas Hofmann. *Unsupervised Learning by Probabilistic Latent Semantic Analysis*. Machine Learning, 2001

# Taxonomy of Classic IR Models



# Two Perspectives for IR Models

- Matching Strategy

- Literal term matching

- Vector Space Model (VSM), Hidden Markov Model (HMM), Language Model (LM)

- Concept matching

- Latent Semantic Indexing (LSI), Probabilistic Latent Semantic Indexing (PLSI), Topical Mixture Model (TMM)

- Learning Capability

- Term weight, query expansion, document expansion, etc

- Vector Space Model, Latent Semantic Indexing

- Solid statistical foundations

- Hidden Markov Model, Probabilistic Latent Semantic Indexing (PLSI), Topical Mixture Model (TMM)

# Two Perspectives for IR Models (cont.)

- Literal Term Matching vs. Concept Matching



香港星島日報篇報導引述軍事觀察家的話表示到二零零五年台灣將完全喪失空中優勢原因是中國大陸戰機不論是數量或是性能上都將超越台灣報導指出中國在大量引進俄羅斯先進武器的同時也得加快研發自製武器系統目前西安飛機製造廠任職的改進型飛豹戰機即將部署尚未與蘇愷三十通道地對地攻擊住宅飛機以督促遇到挫折的監控其戰機目前也已經取得了重大階段性的認知成果根據日本媒體報導在台海戰爭隨時可能爆發情況之下北京方面的基本方針使用高科技答應局部戰爭因此解放軍打算在二零零四年前又有包括蘇愷三十二期在內的兩百架蘇霍伊戰鬥機

# HMM/N-gram-based Model

- Model the query  $Q$  as a sequence of input observations (index terms),  $Q = q_1 q_2 \dots q_n \dots q_N$
- Model the doc  $D$  as a discrete HMM composed of distributions 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,  $Q = q_1 q_2 \dots q_n \dots q_N$ , predicted by the doc  $D$ 
  - A generative model for IR

$$D^* = \arg \max_D P(D \text{ is } R | Q)$$

$$\approx \arg \max_D P(Q | D \text{ is } R) P(D \text{ is } R)$$

$$\approx \arg \max_D P(Q | D \text{ is } R) \quad \text{with the assumption that .....}$$

# HMM/N-gram-based Model (cont.)

- Given a word sequence,  $W$ , of length  $N$

$$\Rightarrow W = w_1 w_2 \dots w_n \dots w_N$$

- How to estimate its corresponding probability ?

$$P(W)$$

$$= P(w_1 w_2 \dots w_n \dots w_N)$$

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

chain rule is applied



Too complicate to estimate all the necessary probability items !

# HMM/N-gram-based Model (cont.)

- *N*-gram approximation (Language Model)

- Unigram

$$P(W) = P(w_1)P(w_2)P(w_3)\dots P(w_N)$$

- Bigram

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

- Trigram

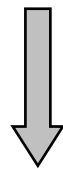
$$P(W) = P(w_1)P(w_2|w_1)P(w_3|w_1w_2)\dots P(w_N|w_{N-2}w_{N-1})$$

- .....

# HMM/N-gram-based Model (cont.)

- A discrete HMM composed of distributions of  $N$ -gram parameters (viewed as a language model source)

$$P(Q|D \text{ is } R) = P(q_1|D) \prod_{n=2}^N P(q_n|q_{n-1}, D) \quad \text{bigram modeling}$$

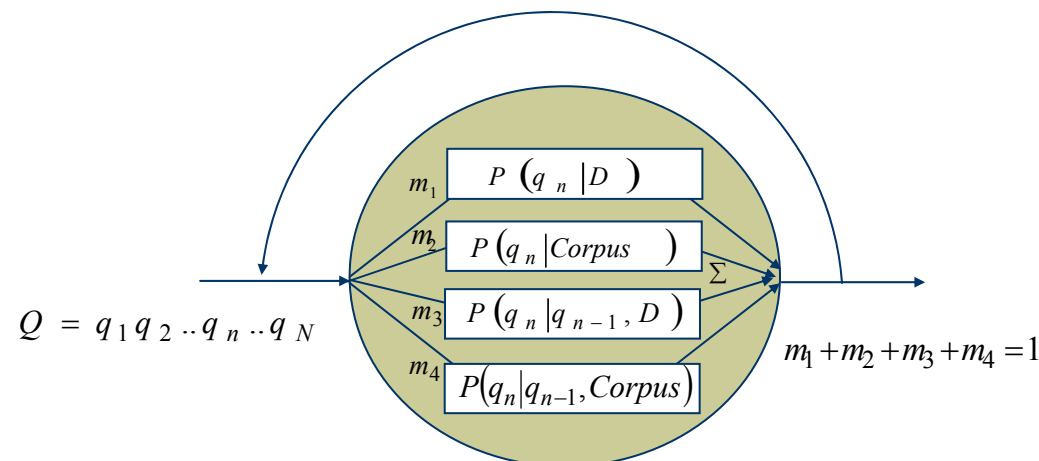


**smoothing/interpolation**

, but reasons for what: avoiding zero prob., and ...?

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

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



A mixture of  $N$  probability distributions



# HMM/N-gram-based Model (cont.)

- **Variants:** Three Types of HMM Structures

- Type I: Unigram-Based (Uni)

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

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

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

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

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

$P(\text{陳水扁 總統 視察 阿里山 小火車}|D)$   
 $= [m_1 P(\text{陳水扁}|D) + m_2 P(\text{陳水扁}|C)] \times [m_1 P(\text{總統}|D) + m_2 P(\text{總統}|C) + m_3 P(\text{總統}|\text{陳水扁}, D) + m_4 P(\text{總統}|\text{陳水扁}, C)]$   
 $\times [m_1 P(\text{視察}|D) + m_2 P(\text{視察}|C) + m_3 P(\text{視察}|\text{總統}, D) + m_4 P(\text{視察}|\text{總統}, C)] \times \dots$

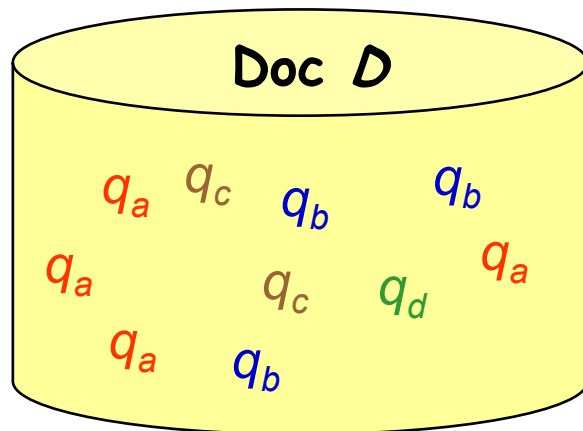
# HMM/N-gram-based Model (cont.)

- The role of the corpus  $N$ -gram probabilities
  - Model the general distribution of the index terms
    - Help to solve zero-frequency problem
    - Help to differentiate the contributions of different missing terms in a doc (global information like IDF?)
  - The corpus  $N$ -gram probabilities were estimated using an outside corpus

$$P(q_n | Corpus)$$

$$P(q_n | q_{n-1}, Corpus)$$

$$P(q_n | D) = 0!$$



$$P(q_a | D) = 0.4$$

$$P(q_b | D) = 0.3$$

$$P(q_c | D) = 0.2$$

$$P(q_d | D) = 0.1$$

$$P(q_e | D) = 0.0$$

$$P(q_f | D) = 0.0$$

# HMM/N-gram-based Model (cont.)

- 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  $D$

Or number of terms in 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  $D$

- Similar formulas for corpus  $N$ -grams

Counts of term  $q_i$  in the *Corpus*

$$P(q_i|Corpus) = \frac{C_{Corpus}(q_i)}{|Corpus|}$$

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

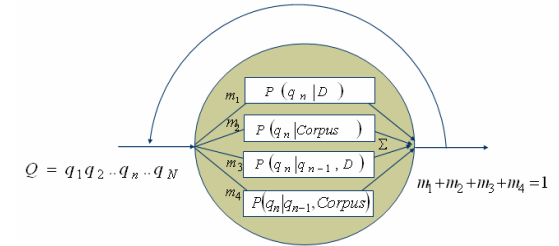
**Corpus:** an outside corpus or just the doc collection

# HMM/N-gram-based Model (cont.)

- Basically,  $m_1, m_2, m_3, m_4$ , can be estimated by using the Expectation-Maximization (EM) algorithm
  - All docs share the same weights  $m_i$  here
  - The  $N$ -gram probability distributions also can be estimated using the EM algorithm instead of the maximum likelihood (ML) estimation
    - Unsupervised: using doc itself, ML
    - Supervised: using query exemplars, EM
- 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

because of the insufficiency of training data

# HMM/N-gram-based Model (cont.)



- Expectation-Maximization Training

- The weights are tied among the documents

- E.g.  $m_1$  of **Type I HMM** can be trained using the following equation:

$$\hat{m}_1 = \frac{\sum_{Q \in [TrainSet]_Q} \sum_{D \in [Doc]_{R \text{ to } Q}} \sum_{q_n \in Q} \left[ \frac{m_1 P(q_n | D)}{m_1 P(q_n | D) + m_2 P(q_n | Corpus)} \right]}{\sum_{Q \in [TrainSet]_Q} |Q| \cdot |[Doc]_{R \text{ to } Q}|}$$

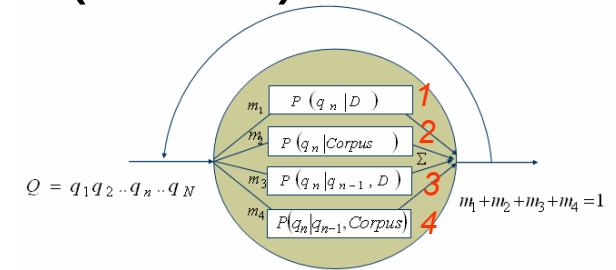
819 queries    ≤ 2265 docs

the new weight →  $\hat{m}_1$       the old weight →  $m_1 P(q_n | D)$

- 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  $|[Doc]_{R \text{ to } Q}|$  is the total number of docs relevant to the query  $Q$

# HMM/N-gram-based Model (cont.)

- Expectation-Maximization Training
  - Step 1: Expectation



$$P(\mathbf{Q}, \mathbf{K} | \hat{D}) = P(\mathbf{K} | \mathbf{Q}, \hat{D}) P(\mathbf{Q} | \hat{D})$$

query word sequence      mixture sequence

$$\mathbf{Q} = q_1 q_2 \cdots q_{N-1} q_N$$

$$\mathbf{K} = k_1 k_2 \cdots k_{N-1} k_N$$

- Log-likelihood expression and take expectation over  $\mathbf{K}$

Take expectation on all possible mixture sequences  $\mathbf{K}$  (conditioned on  $\mathbf{Q}, D$ )

$$\log P(\mathbf{Q}, \mathbf{K} | \hat{D}) = \log P(\mathbf{K} | \mathbf{Q}, \hat{D}) + \log P(\mathbf{Q} | \hat{D})$$

$$\Rightarrow \log P(\mathbf{Q} | \hat{D}) = \log P(\mathbf{Q}, \mathbf{K} | \hat{D}) - \log P(\mathbf{K} | \mathbf{Q}, \hat{D})$$

$$\Rightarrow E[\log P(\mathbf{Q} | \hat{D})]_{\mathbf{K} | \mathbf{Q}, D} = E[\log P(\mathbf{Q}, \mathbf{K} | \hat{D}) - \log P(\mathbf{K} | \mathbf{Q}, \hat{D})]_{\mathbf{K} | \mathbf{Q}, D}$$

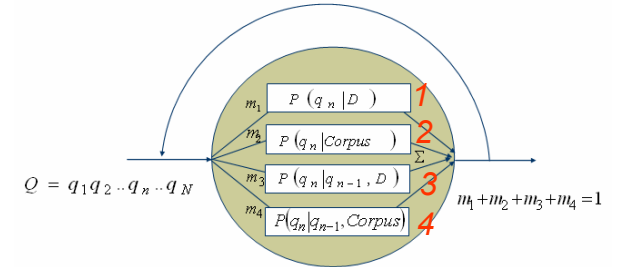
$$\Rightarrow \sum_{\mathbf{K}} P(\mathbf{K} | \mathbf{Q}, D) \log P(\mathbf{Q} | \hat{D}) = \sum_{\mathbf{K}} P(\mathbf{K} | \mathbf{Q}, D) (\log P(\mathbf{Q}, \mathbf{K} | \hat{D}) - \log P(\mathbf{K} | \mathbf{Q}, \hat{D}))$$

$$\Rightarrow \log P(\mathbf{Q} | \hat{D}) = \sum_{\mathbf{K}} P(\mathbf{K} | \mathbf{Q}, D) \log P(\mathbf{Q}, \mathbf{K} | \hat{D}) - \sum_{\mathbf{K}} P(\mathbf{K} | \mathbf{Q}, D) \log P(\mathbf{K} | \mathbf{Q}, \hat{D})$$

# HMM/N-gram-based Model (cont.)

- Explanation**

$M$  mixtures of distributions



$$P(\mathbf{Q} | \hat{D}) = \prod_{n=1}^N \sum_{k_n=1}^M (m_{k_n} P(q_n | k_n, \hat{D}))$$

Note

$$\prod_{k=1}^M (\sum_{i=1}^M a_{ki})$$

$$= (a_{11} + a_{12} + \dots + a_{1M})(a_{21} + a_{22} + \dots + a_{2M}) \dots (a_{M1} + a_{M2} + \dots + a_{MM})$$

$$= \sum_{i_1=1}^M \sum_{i_2=1}^M \dots \sum_{i_M=1}^M \prod_{k=1}^M a_{ki_{k}}$$

Sum-product  $\rightarrow$  product-sum

$$= (m_1 P(q_1 | k_1, \hat{D}) + \dots + m_M P(q_1 | k_M, \hat{D})) \times (m_1 P(q_2 | k_1, \hat{D}) + \dots + m_M P(q_2 | k_M, \hat{D}))$$

$$\times \dots \times (m_1 P(q_N | k_1, \hat{D}) + \dots + m_M P(q_N | k_M, \hat{D}))$$

$$= \sum_{k_1=1}^M \sum_{k_2=1}^M \dots \sum_{k_N=1}^M \left[ \prod_{n=1}^N m_{k_n} P(q_n | k_n, \hat{D}) \right] \text{ where } m_{k_n} = P(k_n | \hat{D})$$

$$\Rightarrow P(\mathbf{Q} | \hat{D}) = \sum_{k_1=1}^M \sum_{k_2=1}^M \dots \sum_{k_N=1}^M \left[ \prod_{n=1}^N P(k_n | \hat{D}) P(q_n | k_n, \hat{D}) \right]$$

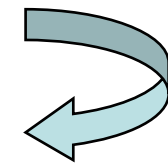
$$\mathbf{Q} = q_1 q_2 \dots q_{N-1} q_N$$

$$\mathbf{K} = k_1 k_2 \dots k_{N-1} k_N$$

$$= \sum_{k_1=1}^M \sum_{k_2=1}^M \dots \sum_{k_N=1}^M \left[ \prod_{n=1}^N P(q_n, k_n | \hat{D}) \right]$$

$$= \sum_{k_1=1}^M \sum_{k_2=1}^M \dots \sum_{k_N=1}^M \left[ P(q_1, k_1, q_2, k_2, \dots, q_N, k_N | \hat{D}) \right]$$

$$= \sum_{\mathbf{K}} \left[ P(\mathbf{Q}, \mathbf{K} | \hat{D}) \right]$$



Independence Assumption

How many kinds of  $\mathbf{K}$ ? ( $M^N$  kinds)

# HMM/N-gram-based Model (cont.)

- Expectation-Maximization Training

- Step 1: Expectation (cont.)

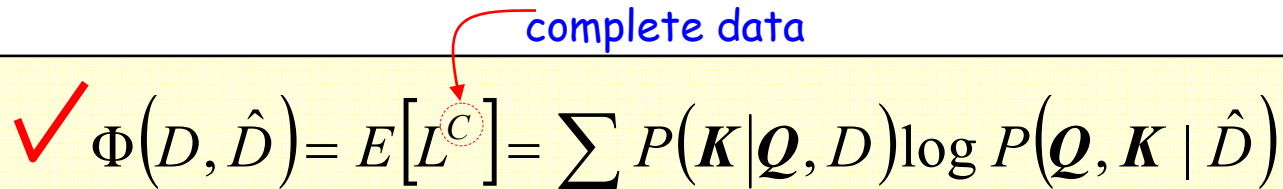
- Express  $\log P(\mathbf{Q} | \hat{D})$  using two auxiliary functions

$$\log P(\mathbf{Q} | \hat{D}) = \sum_{\mathbf{K}} P(\mathbf{K} | \mathbf{Q}, D) \log P(\mathbf{Q}, \mathbf{K} | \hat{D}) - \sum_{\mathbf{K}} P(\mathbf{K} | \mathbf{Q}, D) \log P(\mathbf{K} | \mathbf{Q}, \hat{D})$$

where

$$\log P(\mathbf{Q} | \hat{D}) = \Phi(D, \hat{D}) - H(D, \hat{D})$$

complete data


$$\checkmark \Phi(D, \hat{D}) = E[L^C] = \sum_{\mathbf{K}} P(\mathbf{K} | \mathbf{Q}, D) \log P(\mathbf{Q}, \mathbf{K} | \hat{D})$$

$$H(D, \hat{D}) = \sum_{\mathbf{K}} P(\mathbf{K} | \mathbf{Q}, D) \log P(\mathbf{K} | \mathbf{Q}, \hat{D})$$



# HMM/N-gram-based Model (cont.)

- Expectation-Maximization Training
  - Step 1: Expectation (cont.)

- We want

$$\log P(\mathbf{Q} | \hat{D}) \geq \log P(\mathbf{Q} | D)$$

*unknown model setting*

$$\begin{aligned} & \log P(\mathbf{Q} | \hat{D}) - \log P(\mathbf{Q} | D) \\ &= [\Phi(D, \hat{D}) - H(D, \hat{D})] - [\Phi(D, D) - H(D, D)] \\ &= \Phi(D, \hat{D}) - \Phi(D, D) - H(D, \hat{D}) + H(D, D) \end{aligned}$$

$\geq 0$

# HMM/N-gram-based Model (cont.)

- Expectation-Maximization Training

- Step 1: Expectation (cont.)

- $-H(D, \hat{D}) + H(D, D)$  has the following property

$$-H(D, \hat{D}) + H(D, D)$$

$$= -\left[ \sum_{\mathbf{K}} P(\mathbf{K}|\mathbf{Q}, D) \log P(\mathbf{K}|\mathbf{Q}, \hat{D}) \right] + \left[ \sum_{\mathbf{K}} P(\mathbf{K}|\mathbf{Q}, D) \log P(\mathbf{K}|\mathbf{Q}, D) \right]$$

$$= -\sum_{\mathbf{K}} \left[ P(\mathbf{K}|\mathbf{Q}, D) \log \frac{P(\mathbf{K}|\mathbf{Q}, \hat{D})}{P(\mathbf{K}|\mathbf{Q}, D)} \right]$$

$$\sum_x \left[ P(x) \log \frac{P(x)}{q(x)} \right]$$

*Kullback-Leibler (KL) distance*

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

$$\geq \sum_{\mathbf{K}} \left[ P(\mathbf{K}|\mathbf{Q}, D) \left( 1 - \frac{P(\mathbf{K}|\mathbf{Q}, \hat{D})}{P(\mathbf{K}|\mathbf{Q}, D)} \right) \right]$$

*Jensen's inequality*

$$= -\sum_{\mathbf{K}} \left[ P(\mathbf{K}|\mathbf{Q}, D) - P(\mathbf{K}|\mathbf{Q}, \hat{D}) \right]$$

$$= 0$$

# HMM/N-gram-based Model (cont.)

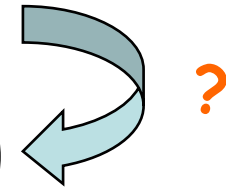
- Expectation-Maximization Training

- Step 1: Expectation (cont.)

- Therefore, for maximizing  $\log P(\mathbf{Q} | \hat{D})$ , we only need to maximize the  $\Phi$ -function (auxiliary function)

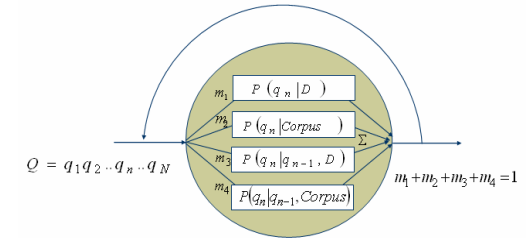
$$\Phi(D, \hat{D}) = \sum_{\mathbf{K}} P(\mathbf{K} | \mathbf{Q}, D) \log P(\mathbf{Q}, \mathbf{K} | \hat{D})$$

- If unigram was used, the  $\Phi$ -function can be further expressed as

$$\begin{aligned} \Phi(D, \hat{D}) &= \sum_{\mathbf{K}} P(\mathbf{K} | \mathbf{Q}, D) \log P(\mathbf{Q}, \mathbf{K} | \hat{D}) \\ &\stackrel{?}{=} \sum_{q_n \in \mathbf{Q}} \sum_k P(k | q_n, D) \log P(q_n, k | \hat{D}) \end{aligned}$$


# HMM/N-gram-based Model (cont.)

- Expectation-Maximization Training
  - Step 1: Expectation (cont.)



Where :

$m_k = P(k|D)$ ,

$\hat{m}_k = P(k|\hat{D})$ .

empirical distribution      the model

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

Auxiliary  
function

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

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

# HMM/N-gram-based Model (cont.)

- Expectation-Maximization Training
  - Step 1: Expectation (cont.)
    - the  $\Phi$ -function (auxiliary function) can be treated in two parts

$$\Phi_m = \sum_{q_n \in \mathcal{Q}} \sum_k \frac{P(q_n|k, D) m_k}{\sum_j P(q_n|j, D) m_j} \log \hat{m}_k$$

$$\Phi_{P(q|k, \hat{D})} = \sum_{q_n \in \mathcal{Q}} \sum_k \frac{P(q_n|k, D) m_k}{\sum_j P(q_n|j, D) m_j} \log P(q_n|k, \hat{D})$$

The reestimation of probabilities  
 $P(q|k, \hat{D})$  will not be discussed here!

# HMM/N-gram-based Model (cont.)

- Expectation-Maximization Training
  - Step 2: Maximization
    - Apply Lagrange Multiplier

By applying Lagrange Multiplier  $\ell$

$$\text{Suppose that } F = \sum_{j=1}^N w_j \log y_j = \sum_{j=1}^N w_j \log y_j + \ell \left( \sum_{j=1}^N y_j - 1 \right)$$

$$\frac{\partial F}{\partial y_j} = \frac{w_j}{y_j} + \ell = 0 \Rightarrow \ell = -\frac{w_j}{y_j} \quad \forall j$$

$$\ell \sum_{j=1}^N y_j = -\sum_{j=1}^N w_j \Rightarrow \ell = -\frac{\sum_{j=1}^N w_j}{\sum_{j=1}^N y_j}$$

$$\therefore y_j = \frac{w_j}{\sum_{j=1}^N w_j}$$

**Constraint**

Note :

$$\frac{\partial \log y_j}{\partial y_j} = \frac{1}{y_j}$$

# HMM/N-gram-based Model (cont.)

- Expectation-Maximization Training
  - Step 2: Maximization (cont.)
    - Apply Lagrange Multiplier

Note :

$$\frac{\partial \log \hat{m}_k}{\partial \hat{m}_k} = \frac{1}{\hat{m}_k}$$

$$\bar{\Phi}_m = \sum_{q_n \in \mathcal{Q}} \sum_k \left\{ \frac{P(q_n|k,D) m_k}{\sum_j P(q_n|j,D) m_j} \log \hat{m}_k \right\} + l \left( \sum_i \hat{m}_i - 1 \right)$$

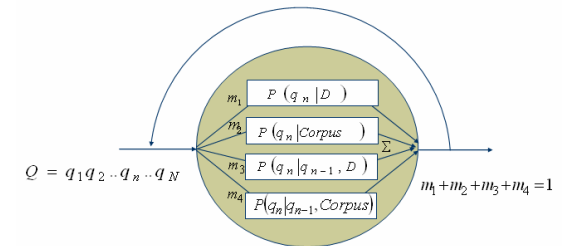
normalization constraints  
using Lagrange multipliers

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

Assume  $G_k = \sum_{q_n \in \mathcal{Q}} \frac{P(q_n|k,D) m_k}{\sum_j P(q_n|j,D) m_j} \Rightarrow \frac{G_1}{\hat{m}_1} = \frac{G_2}{\hat{m}_2} = \dots = \frac{G_k}{\hat{m}_k} = \dots = -l$

# HMM/N-gram-based Model (cont.)

- Expectation-Maximization Training
  - Step 2: Maximization (cont.)



$$\begin{aligned} \therefore l &= - \sum_s G_s && G_k \\ \therefore \hat{m}_k &= \frac{\sum_{q_n \in Q} \frac{P(q_n|k, D) m_k}{\sum_j P(q_n|j, D) m_j}}{\sum_s \sum_{q_n \in Q} \frac{P(q_n|k, D) m_s}{\sum_j P(q_n|j, D) m_j}} && l \end{aligned}$$

– Extension:

- Multiple training queries for a doc
- Weights are tied among docs

$$\hat{m}_k = \frac{\sum_{Q \in [TrainSet]} \sum_{D \in [Doc]} \sum_{q_n \in Q} \left[ \frac{P(q_n|D) m_k}{\sum_j P(q_n|D) m_j} \right]}{\sum_{Q \in [TrainSet]} |Q| \cdot |[Doc]_{R \text{ to } Q}|}$$



# HMM/N-gram-based Model (cont.)

- 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	<b>0.6327</b>	0.6069	0.5427	0.4698	0.5220	0.5718
	TQ/SD	0.5658	<b>0.5702</b>	0.4803	0.4411	0.5011	0.5307
TDT3	TQ/TD	<b>0.6569</b>	0.6542	0.6141	0.5343	0.5970	0.6560
	TQ/SD	0.6308	0.6361	0.5808	0.5177	0.5678	<b>0.6433</b>

- Vector space model

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

- 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 (cont.)

- The EM Algorithm is important to HMMs and other learning techniques
  - Discover new model parameters to maximize the log-likelihood of **incomplete data**  $\log P(\mathbf{O}|\lambda)$  by iteratively maximizing the expectation of log-likelihood from **complete data**  $\log P(\mathbf{O}, \mathbf{S}|\lambda)$
- Example
  - The observable training data  $\mathbf{O}$ 
    - We want to maximize  $P(\mathbf{O}|\lambda)$ ,  $\lambda$  is a parameter vector
  - The hidden (unobservable) data  $\mathbf{S}$ 
    - E.g. the component densities of observable data  $\mathbf{O}$ , or the underlying state sequence in HMMs

$$\Phi(\lambda, \hat{\lambda}) = \sum_{\mathbf{o}} E \left[ \log P(\mathbf{o}, \mathbf{s} | \hat{\lambda}) \right]_{\mathbf{s} | \mathbf{o}, \lambda}$$

# HMM/N-gram-based Model (cont.)

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

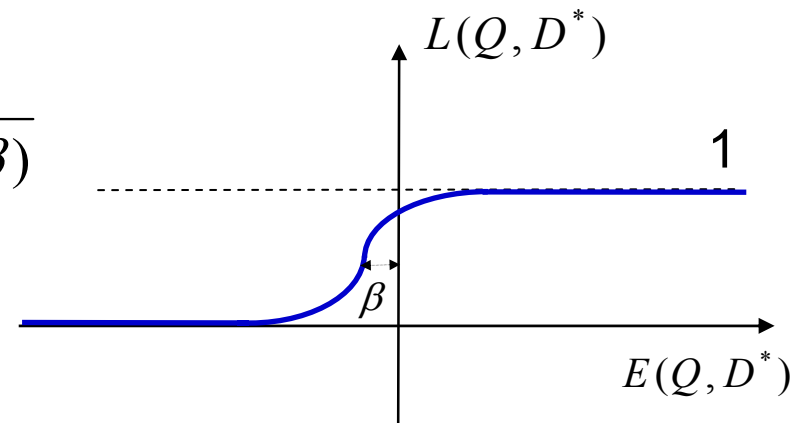
$$E(Q, D^*) = \frac{1}{|Q|} \left[ -\log P(Q|D^* \text{ is } R) + \max_{D'} \log P(Q|D' \text{ is not } R) \right]$$

“>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 (cont.)

- 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 Transformation, (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)

Gradient descent

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

- Where,

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

$$\frac{\partial L(Q, D^*)}{\partial E(Q, D^*)} = \alpha \cdot L(Q, D^*) \cdot [1 - L(Q, D^*)]$$

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

# HMM/N-gram-based Model (cont.)

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

$$\begin{aligned}
 \frac{\partial E(Q, D^*)}{\partial \tilde{m}_1} &= \frac{-1}{|Q|} \frac{\partial \left\{ \sum_{q_n \in Q} \log \left[ \frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P(q_n | D^*) + \frac{e^{\tilde{m}_2}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P(q_n | Corpus) \right] \right\}}{\partial \tilde{m}_1} \\
 &= \frac{-1}{|Q|} \sum_{q_n \in Q} \left\{ \frac{\frac{-e^{\tilde{m}_1}}{(e^{\tilde{m}_1} + e^{\tilde{m}_2})^2} [e^{\tilde{m}_1} P(q_n | D^*) + e^{\tilde{m}_2} P(q_n | Corpus)] + \frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P(q_n | D^*)}{\frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P(q_n | D^*) + \frac{e^{\tilde{m}_2}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P(q_n | Corpus)}} \right\} \\
 &= \frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} - \frac{1}{|Q|} \sum_{q_n \in Q} \left\{ \frac{\frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P(q_n | D^*)}{\frac{e^{\tilde{m}_1}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P(q_n | D^*) + \frac{e^{\tilde{m}_2}}{e^{\tilde{m}_1} + e^{\tilde{m}_2}} P(q_n | Corpus)}} \right\} \\
 &= - \left[ -m_1 + \frac{1}{|Q|} \sum_{q_n \in Q} \frac{m_1 P(q_n | D^*)}{m_1 P(q_n | D^*) + m_2 P(q_n | Corpus)} \right],
 \end{aligned}$$

Note :

$$\begin{aligned}
 [\log f(x)]' &= \frac{1}{f(x)} f'(x) \\
 [f(x)g(x)]' &= f'(x)g(x) + f(x)g'(x) \\
 \left[ \frac{f(x)}{g(x)} \right]' &= \frac{f'(x)g(x) - f(x)g'(x)}{g^2(x)}
 \end{aligned}$$

# HMM/N-gram-based Model (cont.)

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

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

the new weight

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

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

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

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

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 (cont.)

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

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

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

- $m_2$  can be updated in the similar way



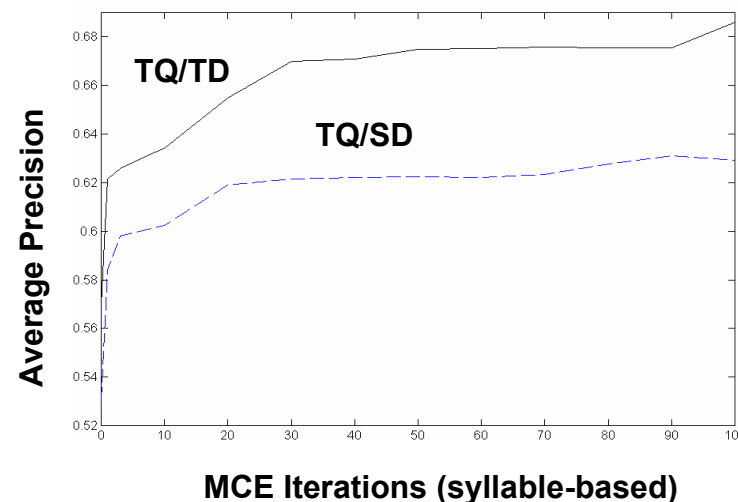
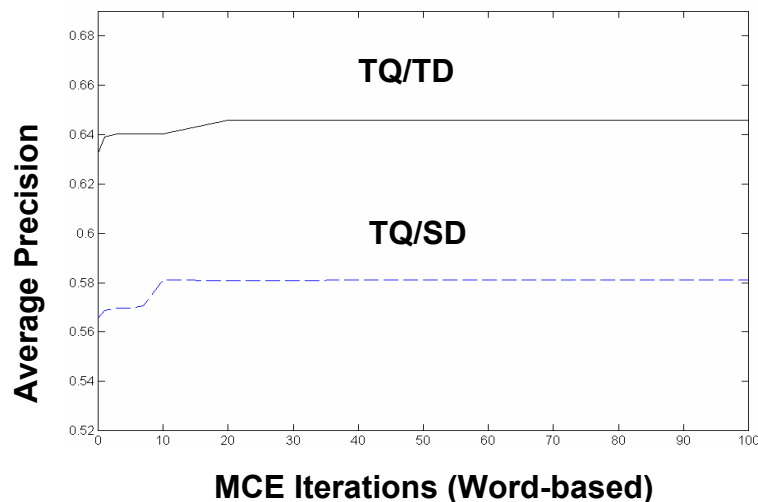
# HMM/N-gram-based Model (cont.)

- 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



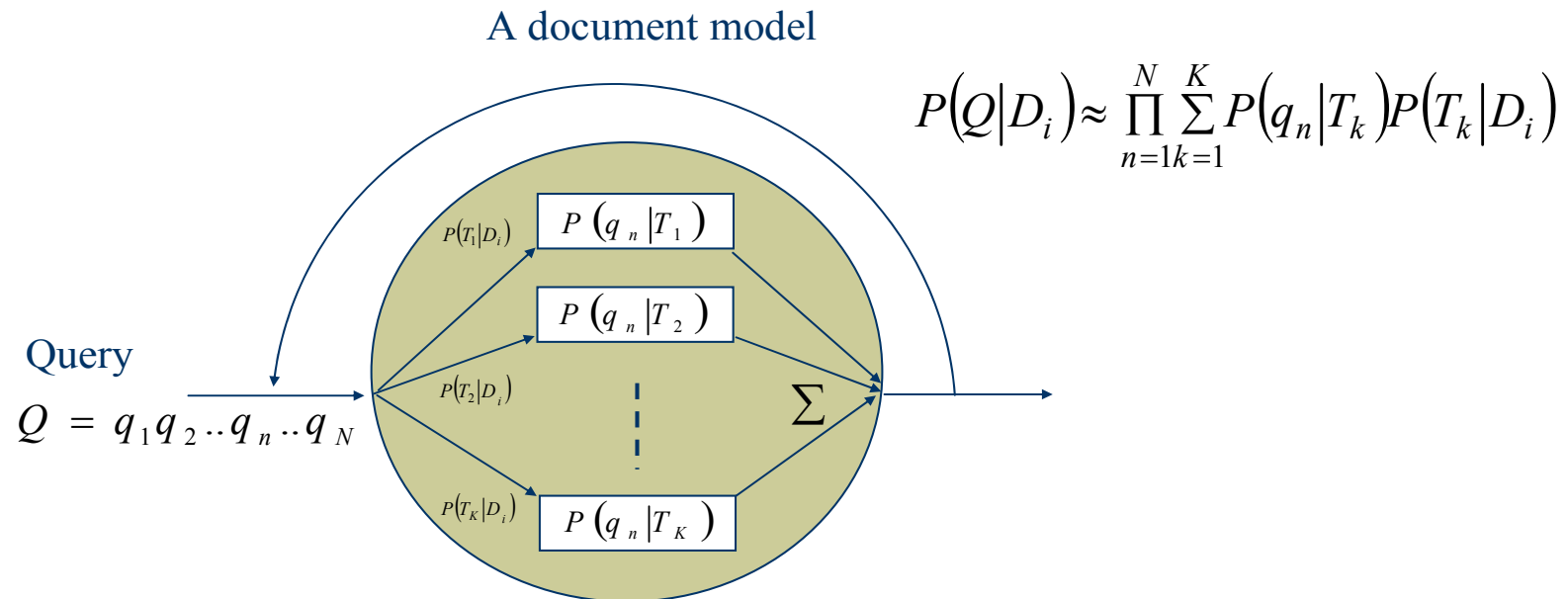
- The results for the syllable-level index features were significantly improved

# HMM/N-gram-based Model (cont.)

- 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

# Topical Mixture Model (TMM)

- Perform Concept Matching in the Likelihood Space (under the likelihood criterion)
  - Latent topical distributions are shared (tied) among docs
- Various Theoretically Attractive Model Training Algorithms can be applied
  - Maximum likelihood (e.g. EM) or discriminative (e.g. MCE or MMI) training



# Topical Mixture Model (cont.)

- EM: Supervised Training
  - Given a training set of query exemplars with the corresponding query-document relevance information

$$\hat{P}(q_n | T_k) = \frac{\sum_{Q \in [\text{TrainSet}]_Q} \sum_{D_i \in [\text{Doc}]_{R \text{ to } Q}} n(q_n, Q) P(T_k | q_n, D_i)}{\sum_{Q \in [\text{TrainSet}]_Q} \sum_{D_i \in [\text{Doc}]_{R \text{ to } Q}} \sum_{q_s \in Q} n(q_s, Q) P(T_k | q_n, D_i)}$$

$$\hat{P}(T_k | D_i) = \frac{\sum_{\substack{Q \in [\text{TrainSet}]_Q \\ \text{st. } D_i \in [\text{DOC}]_{R \text{ to } Q}}} \sum_{q_s \in Q} n(q_s, Q) P(T_k | q_s, D_i)}{\sum_{\substack{Q \in [\text{TrainSet}]_Q \\ \text{st. } D_i \in [\text{DOC}]_{R \text{ to } Q}}} |Q|}$$

, where

$$P(T_k | q_n, D_i) = \frac{P(T_k | D_i) P(q_n | T_k)}{\sum_{l=1}^K P(T_l | D_i) P(q_n | T_l)}$$

# Topical Mixture Model (cont.)

- EM: Unsupervised Training
  - Use each document itself as a query exemplar to train its own document mixture model

$$\hat{P}(w_n | T_k) = \frac{\sum_{D_i \in [D]} n(w_n, D_i) P(T_k | w_n, D_i)}{\sum_{D_i \in [D]} \sum_{w_s \in D_i} n(w_s, D_i) P(T_k | w_s, D_i)}$$

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

, where

$$P(T_k | q_n, D_i) = \frac{P(T_k | D_i) P(w_n | T_k)}{\sum_{l=1}^K P(T_l | D_i) P(w_n | T_l)}$$

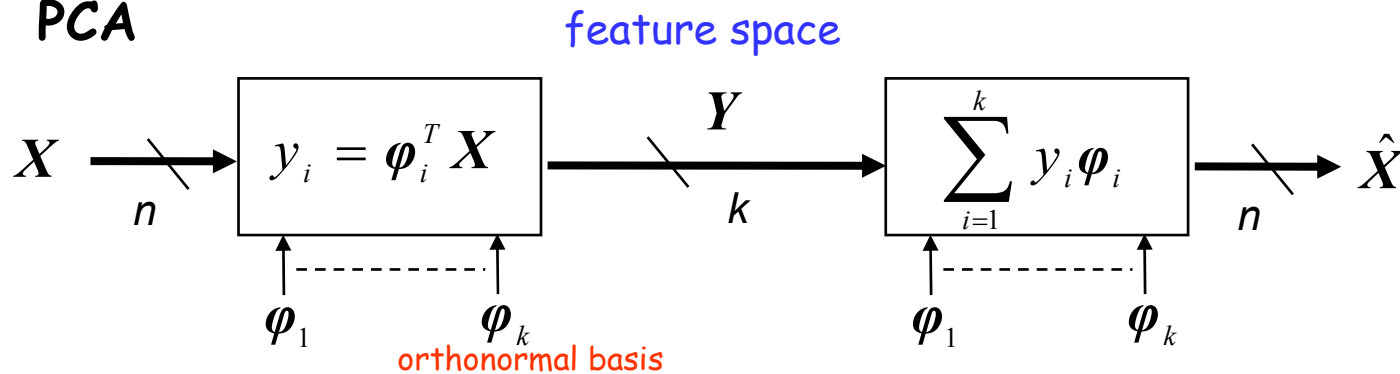
# Latent Semantic Indexing (LSI)

- LSI: a technique that 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 (cont.)

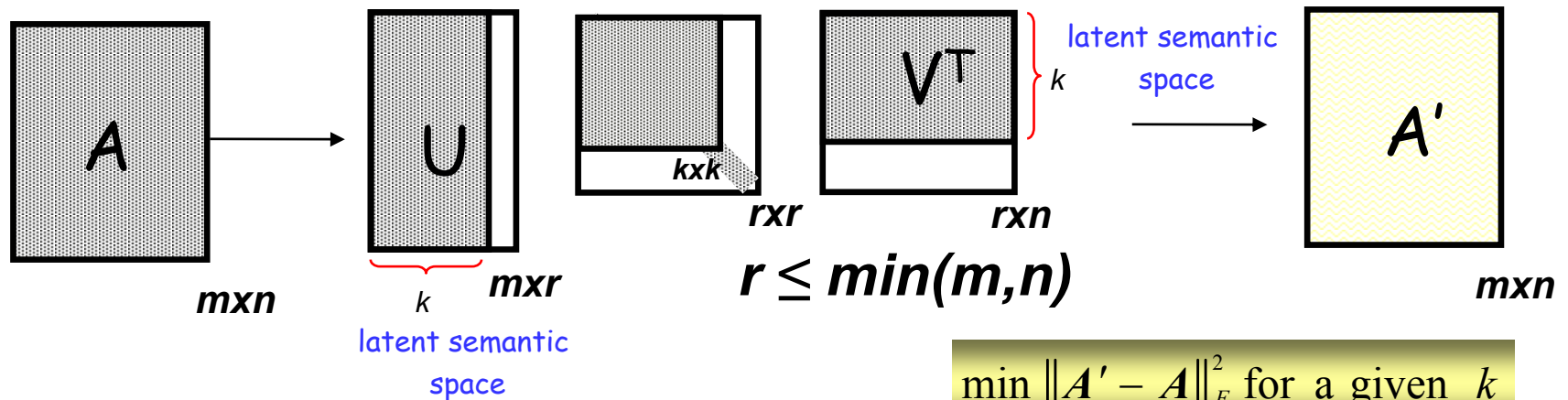
- Dimension Reduction and Feature Extraction

- PCA



- SVD (in LSI)

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



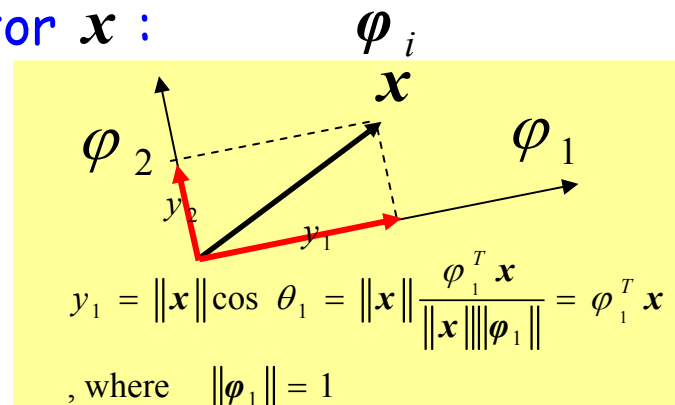
$$\min \|A' - A\|_F^2 \text{ for a given } k$$

# Latent Semantic Indexing (cont.)

- 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

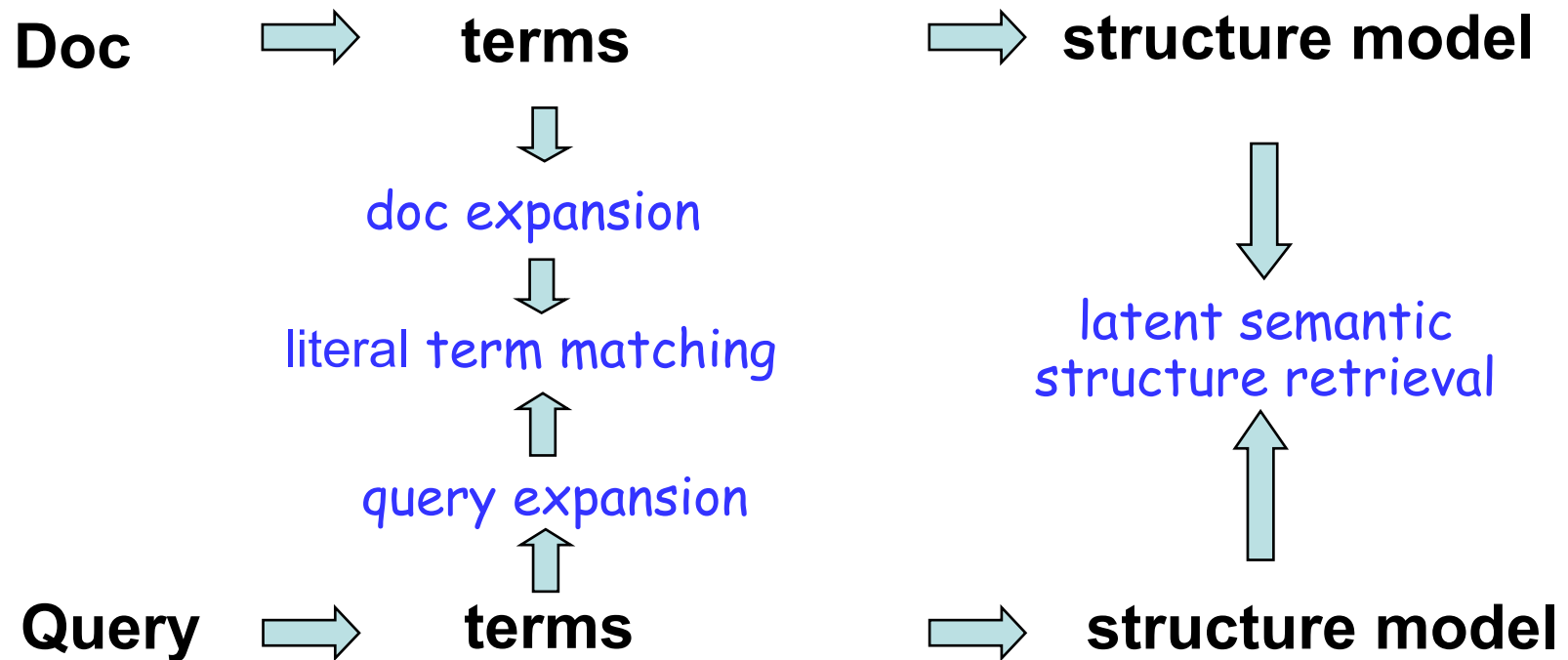
Projection of a Vector  $\mathbf{x}$  :





# Latent Semantic Indexing (cont.)

- Frameworks to circumvent vocabulary mismatch



# Latent Semantic Indexing (cont.)

## 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
<i>human</i>	1	0	0	1	0	0	0	0	0
<i>interface</i>	1	0	1	0	0	0	0	0	0
<i>computer</i>	1	1	0	0	0	0	0	0	0
<i>user</i>	0	1	1	0	1	0	0	0	0
<i>system</i>	0	1	1	2	0	0	0	0	0
<i>response</i>	0	1	0	0	1	0	0	0	0
<i>time</i>	0	1	0	0	1	0	0	0	0
<i>EPS</i>	0	0	1	1	0	0	0	0	0
<i>survey</i>	0	1	0	0	0	0	0	0	1
<i>trees</i>	0	0	0	0	0	1	1	1	0
<i>graph</i>	0	0	0	0	0	0	1	1	1
<i>minors</i>	0	0	0	0	0	0	0	1	1

# Latent Semantic Indexing (cont.)

2-D Plot of Terms and Docs from Example

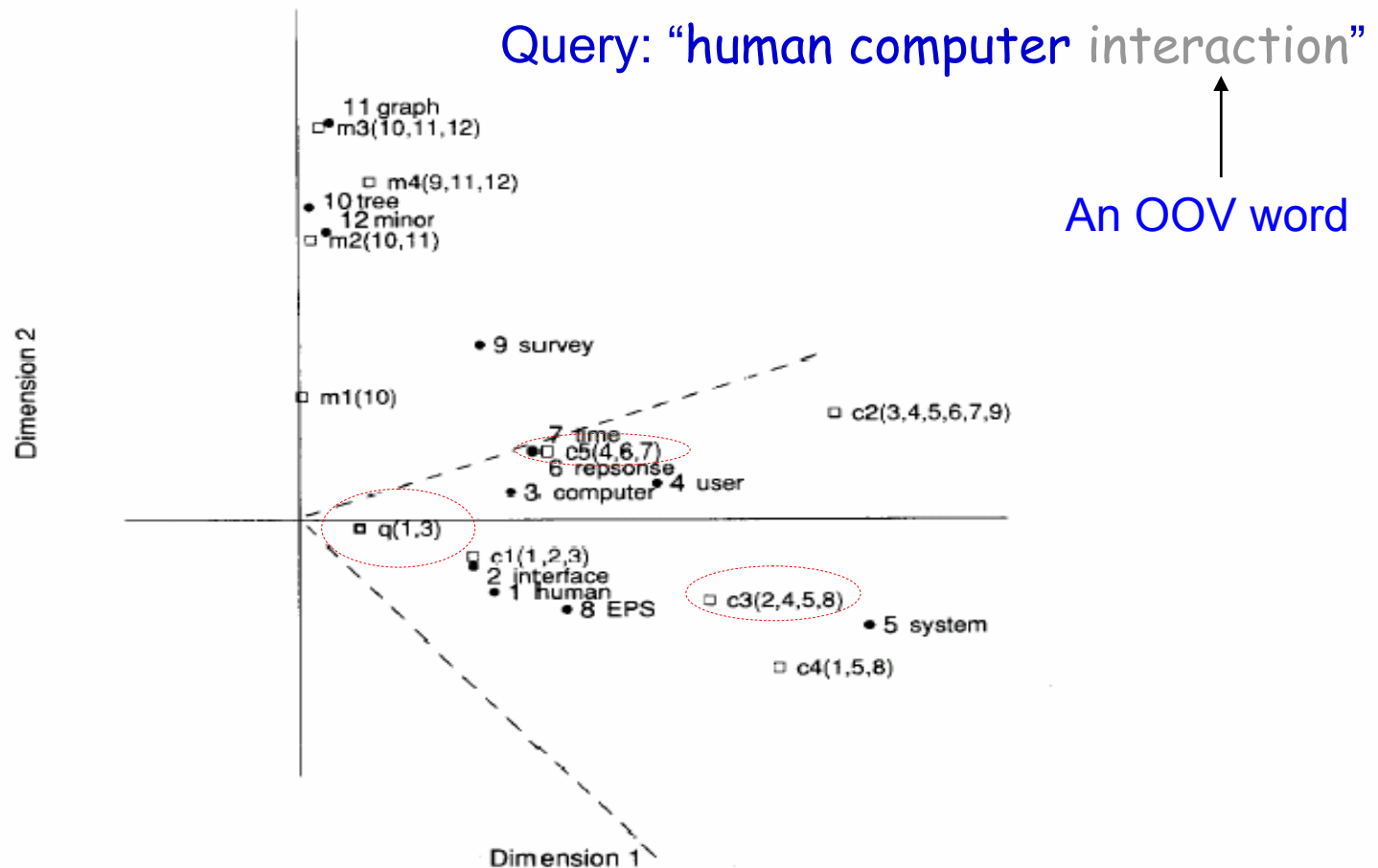
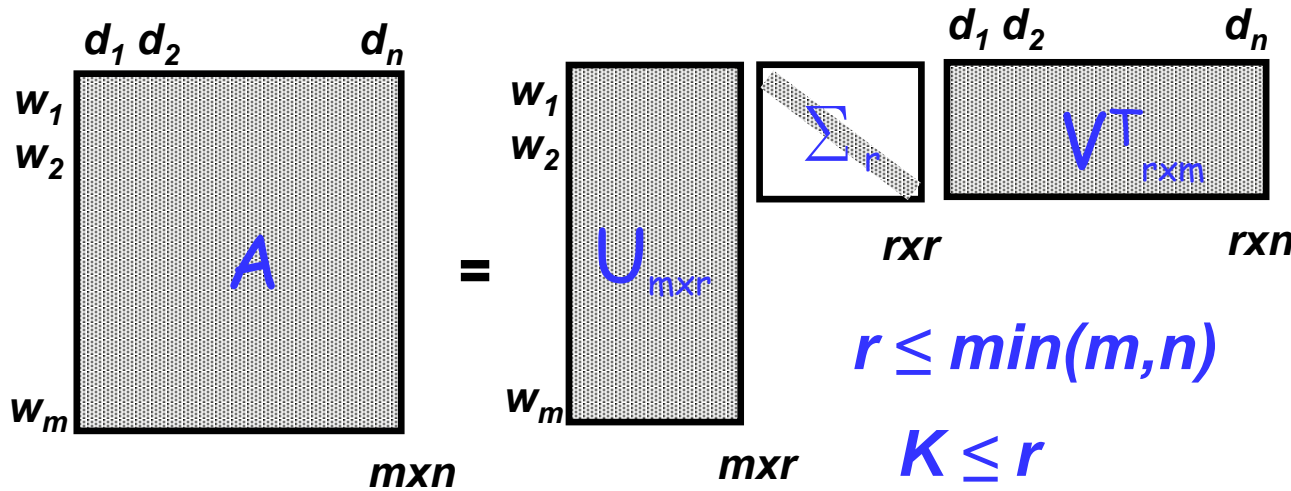


FIG. 1. A two-dimensional plot of 12 Terms and 9 Documents from the same TM set. Terms are represented by filled circles. Documents are shown as open squares, and component terms are indicated parenthetically. The query ("human computer interaction") is represented as a pseudo-document at point  $q$ . 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 (cont.)

- Singular Value Decomposition (SVD)

Row  $A \in R^n$   
Col  $A \in R^m$



Both  $U$  and  $V$  has orthonormal column vectors

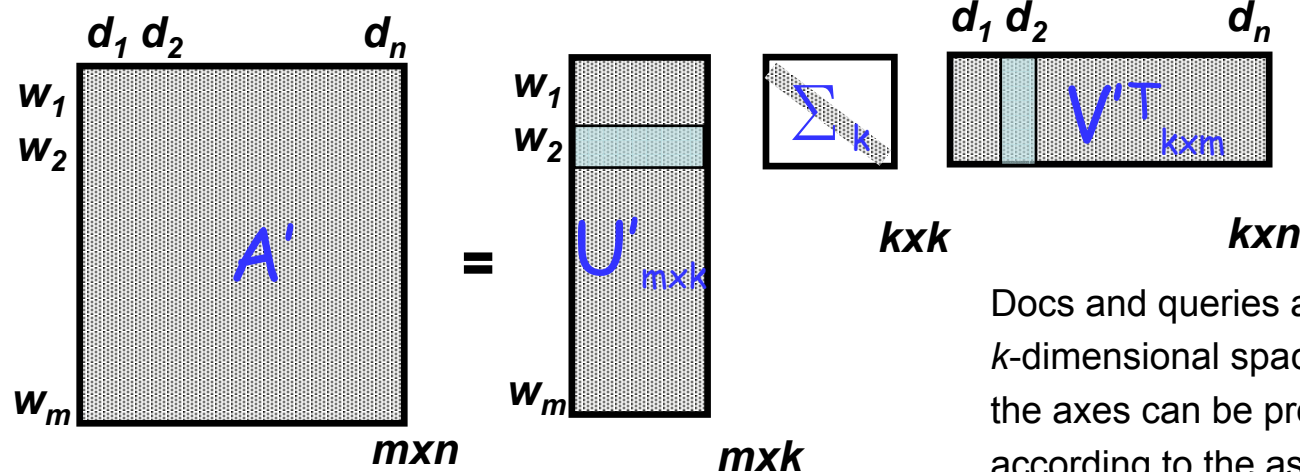
$$U^T U = I_{r \times r}$$

$$V^T V = I_{r \times r}$$

$$r \leq \min(m, n)$$

$$K \leq r$$

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



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

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  $\Sigma_k$

# Latent Semantic Indexing (cont.)

- Singular Value Decomposition (SVD)

- $A^T A$  is symmetric  $n \times n$  matrix

- All eigenvalues  $\lambda_j$  are nonnegative real numbers

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n \geq 0 \quad \Sigma^2 = \text{diag}(\lambda_1, \lambda_1, \dots, \lambda_n)$$

- All eigenvectors  $v_j$  are orthonormal ( $\in \mathbb{R}^n$ )

$$V = [v_1 \ v_2 \ \dots \ v_n] \quad v_j^T v_j = 1 \quad (V^T V = I_{n \times n})$$

- Define **singular values**: sigma  $\sigma_j = \sqrt{\lambda_j}$ ,  $j = 1, \dots, n$

- As the square roots of the eigenvalues of  $A^T A$

- As the lengths of the vectors  $Av_1, Av_2, \dots, Av_n$

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

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

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

.....

$$\|Av_i\|^2 = v_i^T A^T Av_i = v_i^T \lambda_i v_i = \lambda_i$$

$$\Rightarrow \|Av_i\| = \sigma_i$$

# Latent Semantic Indexing (cont.)

- $\{Av_1, Av_2, \dots, Av_r\}$  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  $r \leq n$

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

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

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

$V$ : an orthonormal matrix

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

Known in advance

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

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

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

$$\Rightarrow U \Sigma = AV \Rightarrow U \Sigma V^T = A \underbrace{V V^T}_{I_{n \times n}}$$

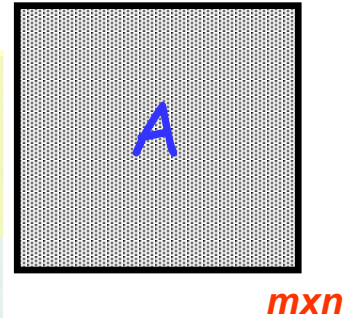
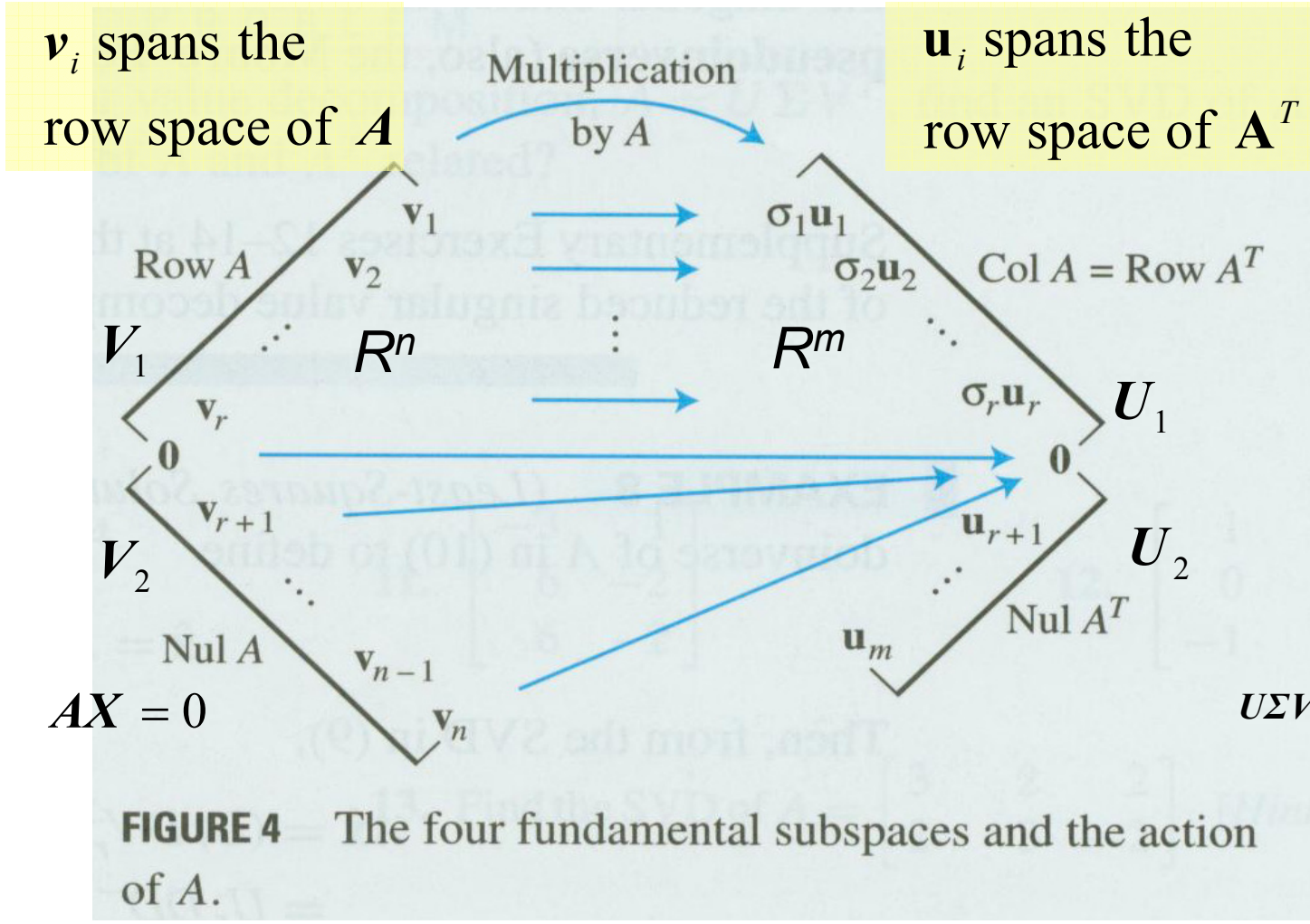
$$\Rightarrow A = U \Sigma V^T$$

$$\Sigma_{m \times m} = \begin{pmatrix} \Sigma_r & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}$$

$$I_{n \times n} \quad ?$$

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

# Latent Semantic Indexing (cont.)

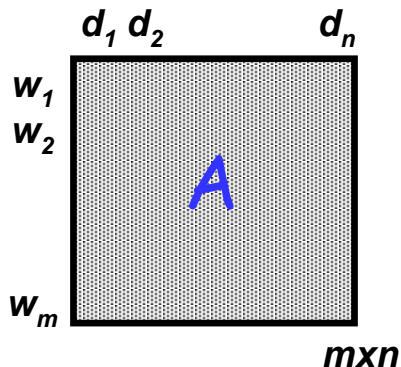


$$\begin{aligned}
 U \Sigma V^T &= \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} V_1^T \\ V_2^T \end{pmatrix} \\
 &= U_1 \Sigma_1 V_1^T \\
 &= A V_1 V_1^T \\
 &= A
 \end{aligned}$$

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

# Latent Semantic Indexing (cont.)

- 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  $A^T A$
- compare a term and a doc  $\rightarrow$  each individual entry of A

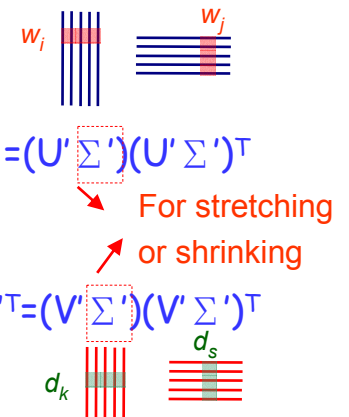
## – The new word-document matrix (A')

$$U' = U_{m \times k}$$

$$\Sigma' = \Sigma_k$$

$$V' = V_{n \times k}$$

- compare two terms  $A'A^T = (U' \Sigma' V'^T)(U' \Sigma' V'^T)^T = U' \Sigma' V'^T V' \Sigma'^T U'^T = (U' \Sigma') (U' \Sigma')^T$ 
  - $\rightarrow$  dot product of two rows of  $U' \Sigma'$
- compare two docs  $A^T A = (U' \Sigma' V'^T)^T (U' \Sigma' V'^T) = V' \Sigma'^T U'^T U' \Sigma' V'^T = (V' \Sigma') (V' \Sigma')^T$ 
  - $\rightarrow$  dot product of two rows of  $V' \Sigma'$
- compare a query word and a doc  $\rightarrow$  each individual entry of A'





# Latent Semantic Indexing (cont.)

- **Fold-in:** find representations for pseudo-docs  $q$ 
  - For objects (new queries or docs) that did not appear in the original analysis
    - Fold-in a new  $m \times 1$  query (or doc) vector

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

Just like a row of  $V$

The separate dimensions are differentially weighted

Query represented by the weighted sum of its constituent term vectors

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

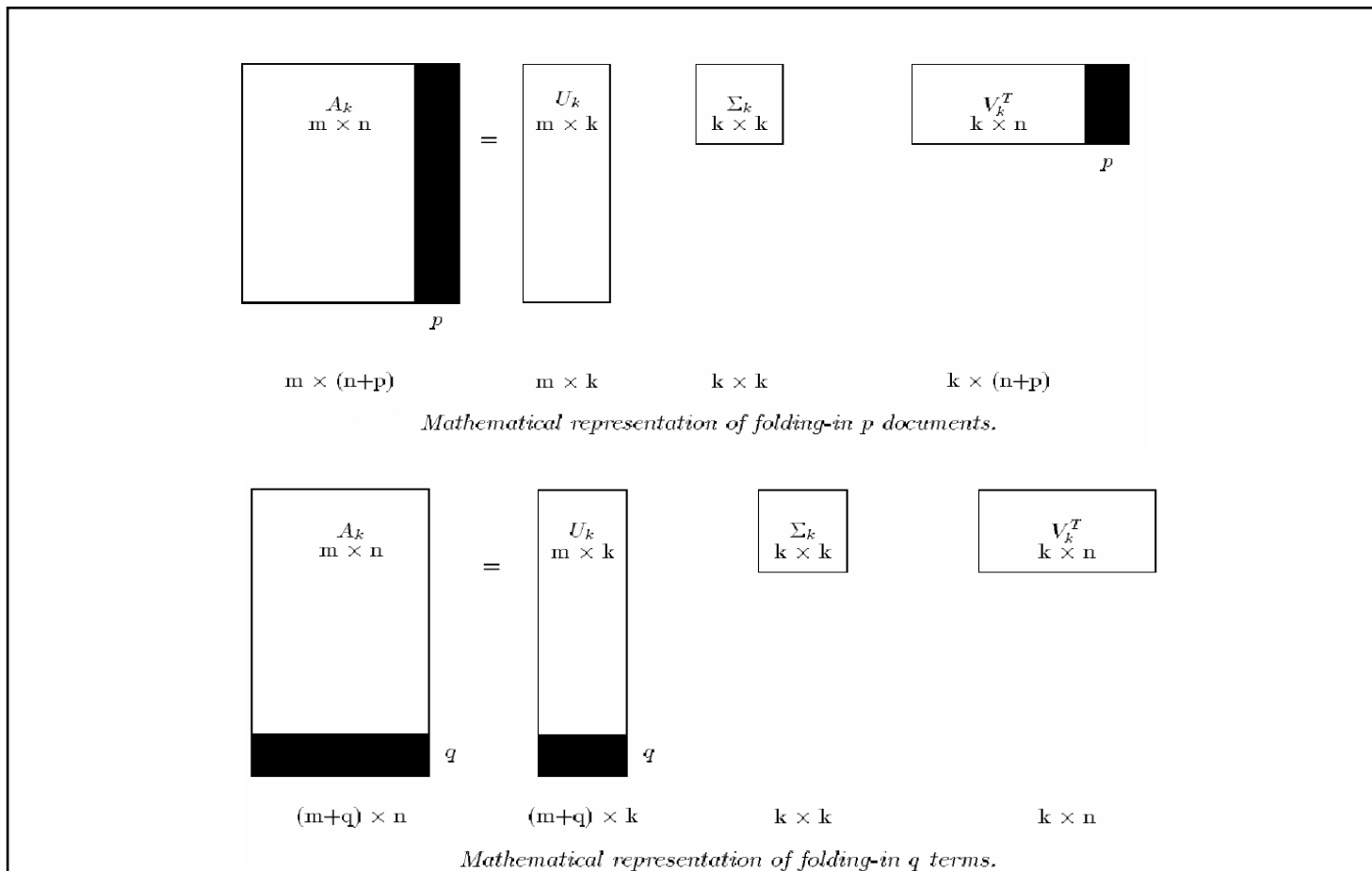
$$\text{sim} \left( \hat{q}, \hat{d} \right) = \text{coine} \left( \hat{q} \Sigma, \hat{d} \Sigma \right) = \frac{\hat{q} \Sigma^2 \hat{d}^T}{\left| \hat{q} \Sigma \right| \left| \hat{d} \Sigma \right|}$$

row vectors

# Latent Semantic Indexing (cont.)

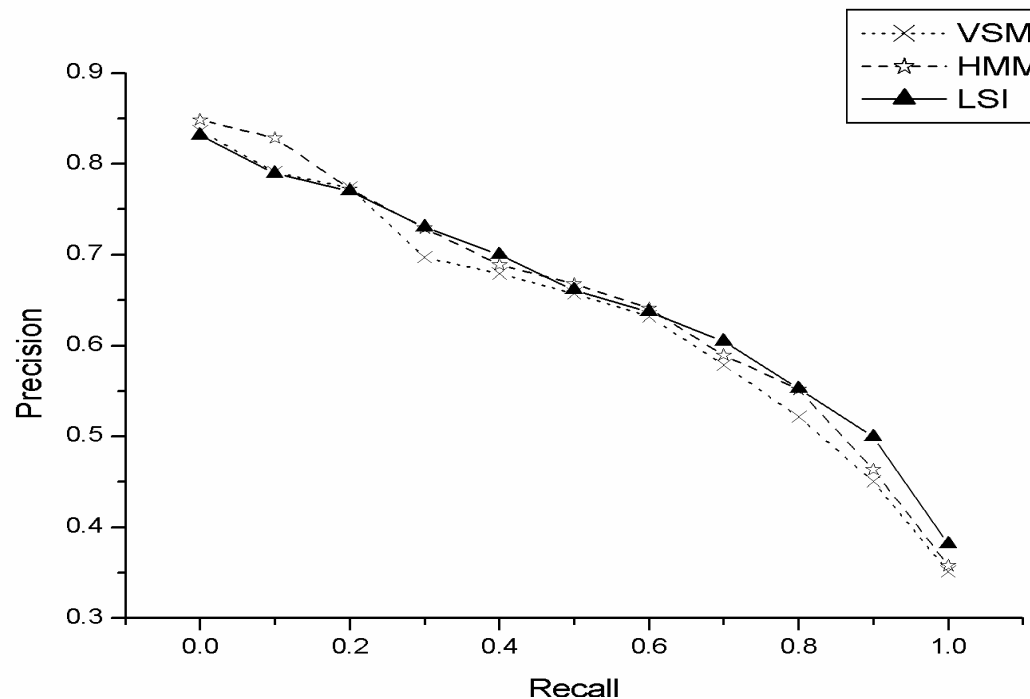
- Fold-in a new 1 x n term vector

$$\hat{t}_{1 \times k} = t_{1 \times n} V_{n \times k} \Sigma_{k \times k}^{-1} \mathbf{1}_{k \times k}$$



# Latent Semantic Indexing (cont.)

- 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)

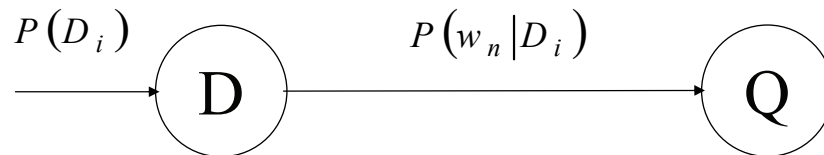
# Latent Semantic Indexing (LSI)

- 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,...

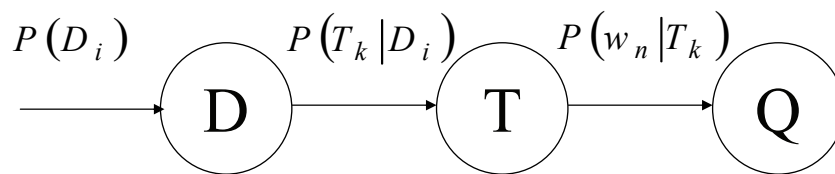
# Probabilistic Latent Semantic Analysis (PLSA)

Thomas Hofmann 1999

- Also called The Aspect Model, Probabilistic Latent Semantic Indexing (PLSI)
  - Graphical Model Representation



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



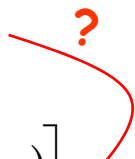
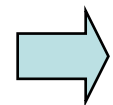
$$\text{sim}(Q, D_i) = P(D_i | Q) = \frac{P(Q, D_i)}{P(Q)} \approx P(Q, D_i) = P(Q | D_i) P(D_i) \approx P(Q | D_i)$$

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

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

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

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



# Probabilistic Latent Semantic Analysis (cont.)

- 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 (cont.)

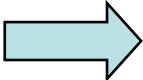
- Assumptions

- **Bag-of-words:** treat docs as *memoryless* source, words are generated independently

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

- **Conditional independent:** the doc  $D_i$  and word  $w_j$  are independent conditioned on the state of the associated latent variable  $T_k$

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



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

Can be viewed as the topics are tied among HMMs

# Probabilistic Latent Semantic Analysis (cont.)

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

- Define the auxiliary function

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

complete data

take expectation

likelihood

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

empirical distribution

the model

- With the property:  $P(w_j, T_k | D_i) \approx P(w_j | T_k) P(T_k | D_i)$

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

without the introduction of query exemplars for training



# Probabilistic Latent Semantic Analysis (cont.)

- Probability estimation using EM (expectation-maximization) algorithm
  - **E** (expectation) step
    - The expression  $\hat{P}(T_k | w_j, D_i)$  can be further decomposed as

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

- The auxiliary function

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

Kullback-Leibler divergence

# Probabilistic Latent Semantic Analysis (cont.)

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

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

normalization constraints using Lagrange multipliers

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

$$\bar{\Phi}_{\hat{P}(T_k|D_i)} = \sum_{w_j} n(w_j, D_i) \sum_{T_k} P(T_k | w_j, D_i) \log \hat{P}(T_k | D_i) + \rho_j \left( 1 - \sum_{T_k} \hat{P}(T_k | D_i) \right)$$

# Probabilistic Latent Semantic Analysis (cont.)

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

The training formula

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

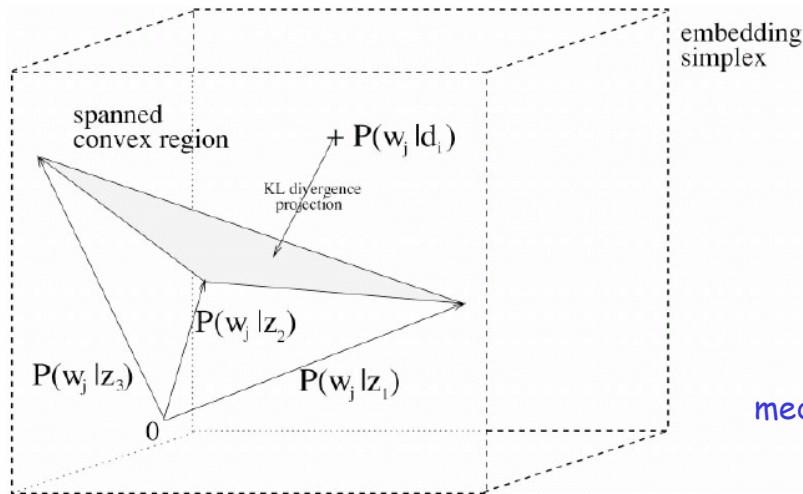
$$\hat{P}(T_k | D_i) = \frac{\sum_{w_j} n(w_j, D_i) P(T_k | w_j, D_i)}{\sum_{T_k} \sum_{w_j} n(w_j, D_i) P(T_k | w_j, D_i)} = \frac{\sum_{w_j} n(w_j, D_i) P(T_k | w_j, D_i)}{\sum_{w_j} n(w_j, D_i)}$$

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

The training formula

# Probabilistic Latent Semantic Analysis (cont.)

- Latent Probability Space



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

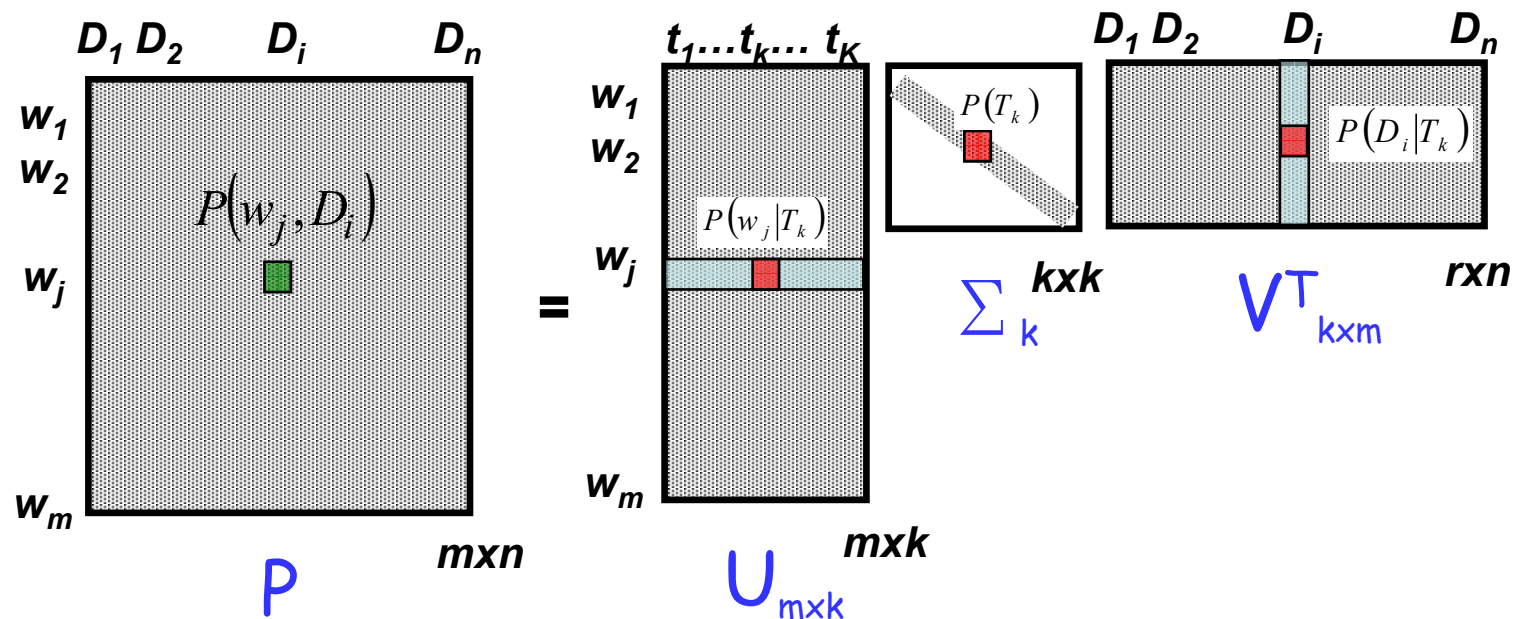
Sketch of the probability simplex and a convex region spanned by class-conditional probabilities in

the aspect model.

$$\begin{aligned}
 P(w_j, D_i) &= \sum_{T_k} P(w_j, T_k, D_i) = \sum_{T_k} P(w_j | T_k, D_i) P(T_k, D_i) \\
 &= \sum_{T_k} P(w_j | T_k) P(T_k) P(D_i | T_k) \\
 P(W, D) &= \hat{U} : (P(w_j | T_k))_{j,k} \cdot \hat{\Sigma} : \text{diag}(P(T_k))_k \cdot \hat{V} : (P(D_i | T_k))_{i,k}
 \end{aligned}$$

# Probabilistic Latent Semantic Analysis (cont.)

- Latent Probability Space



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

# Probabilistic Latent Semantic Analysis (cont.)

- 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 (cont.)

- 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 (cont.)

- 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

$$P_{PLSI-Q^*}(\omega_j | d_i) = \lambda P_{Empirical}(\omega_j | d_i) + (1 - \lambda) P_{PLSA}(\omega_j | d_i)$$
$$P_{Empirical}(\omega_j | d_i) = \frac{n(\omega_j, d_i)}{n(d_i)}$$

Both provide almost identical performance

- It's not known if PLSA was used alone



# Probabilistic Latent Semantic Analysis (cont.)

- Experimental Results

## PLSI-Q\*

- Use the low-dimensional representation  $P(T_k | Q)$  and  $P(T_k | D_i)$  (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 re-weight the different model components (dimensions) by

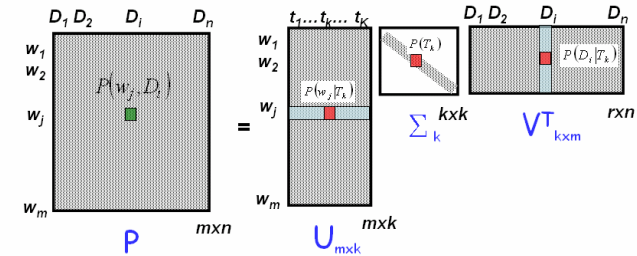
$$R_{PLSI-Q^*}(Q, D_i) = \frac{\sum_k P(T_k | Q) P(T_k | D_i)}{\sqrt{\sum_k P(T_k | Q)^2} \sqrt{\sum_k P(T_k | D_i)^2}}, \text{ where } P(T_k | Q) = \frac{\sum_{q_n \in Q} n(q_n, Q) P(T_k | q_n, Q)}{|Q|}$$

online folded-in

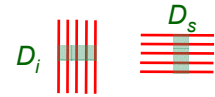
$$\tilde{R}_{PLSI-Q^*}(Q, D_i) = \lambda \cdot R_{PLSI}(Q, D_i) + (1 - \lambda) \cdot R_{VSM}(\bar{Q}, \bar{D}_i)$$

# Probabilistic Latent Semantic Analysis (cont.)

• **Why**  $R_{PLSI-Q^*}(Q, D_i) = \frac{\sum_k P(T_k|Q)P(T_k|D_i)}{\sqrt{\sum_k P(T_k|Q)^2} \sqrt{\sum_k P(T_k|D_i)^2}}$  ?



- Reminder that in LSI, the relations between any two docs can be formulated as



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

$$sim(D_i, D_s) = coine(\hat{D}_i \Sigma, \hat{D}_s \Sigma) = \frac{\hat{D}_i \Sigma^2 \hat{D}_s^T}{|\hat{D}_i \Sigma| |\hat{D}_s \Sigma|}$$

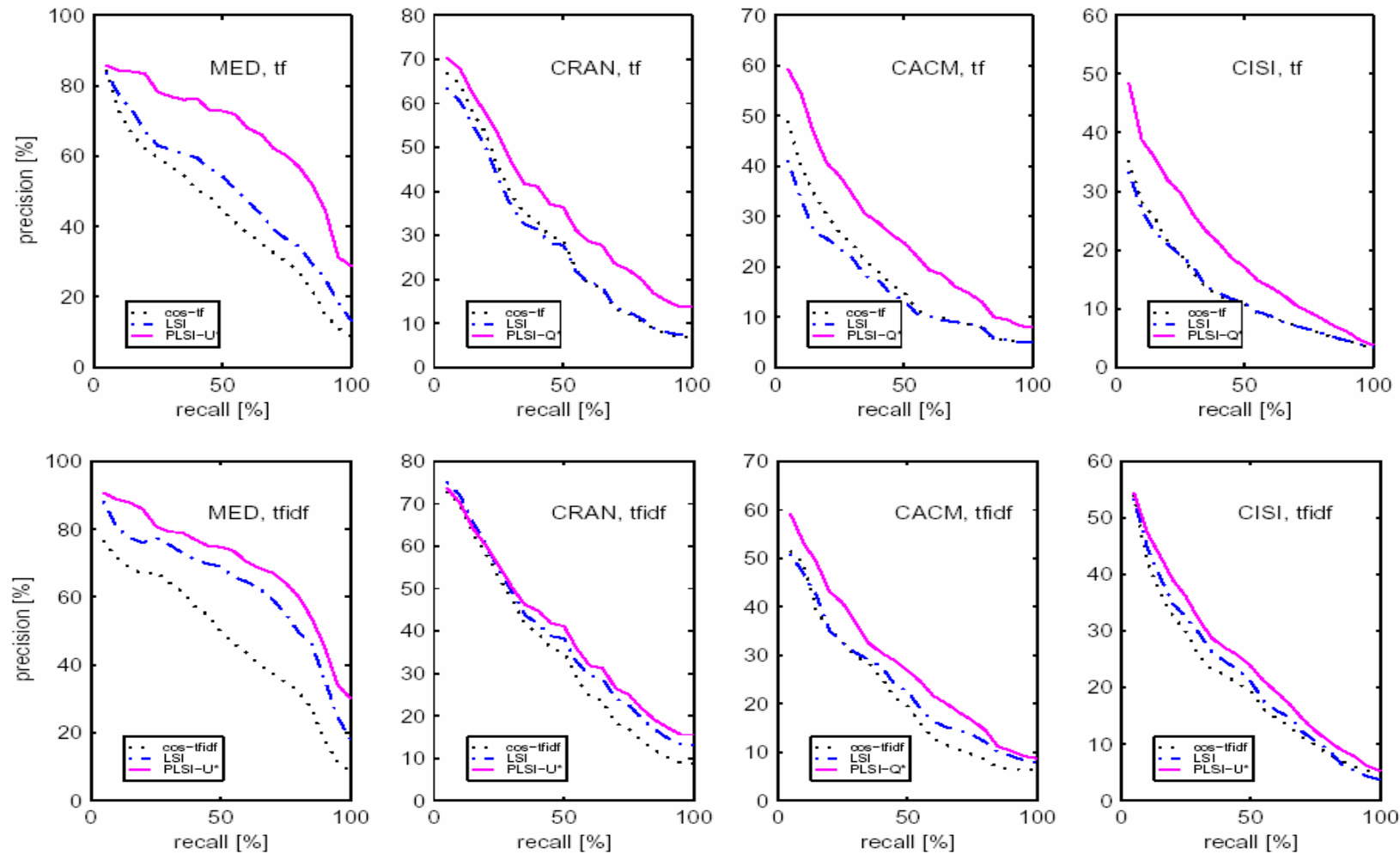
- **PLAS mimics LSI in similarity measure**  $\hat{D}_i$  and  $\hat{D}_s$  are row vectors

$$\begin{aligned}
 R_{PLSI-Q^*}(D_i, D_s) &= \frac{\sum_k P(D_i|T_k)P(T_k)P(T_k)P(D_s|T_k)}{\sqrt{\sum_k [P(D_i|T_k)P(T_k)]^2} \sqrt{\sum_k [P(D_i|T_k)P(T_k)]^2}} \\
 &= \frac{\sum_k P(T_k|D_i)P(D_i)P(T_k|D_s)P(D_s)}{\sqrt{\sum_k [P(T_k|D_i)P(D_i)]^2} \sqrt{\sum_k [P(T_k|D_s)P(D_s)]^2}} \\
 &= \frac{\sum_k P(T_k|D_i)P(T_k|D_s)}{\sqrt{\sum_k P(T_k|D_i)^2} \sqrt{\sum_k P(T_k|D_s)^2}}
 \end{aligned}$$

$P(D_i|T_k)P(T_k) = P(T_k|D_i)P(D_i)$

# Probabilistic Latent Semantic Analysis (cont.)

- Experimental Results



# Comparisons

- TDT-3 Voice of American Spoken Document Collection
  - Measured in *mean* Average Precision (*mAP*)

256 Topics

256 Topics

Retrieval Model	<b>TMM</b>	<b>HMM</b>	PLSA	VSM	LSI
TD	<b>0.7870</b>	<b>0.7174</b>	0.6513	0.6505	0.6440
SD	<b>0.7852</b>	<b>0.7156</b>	0.5989	0.6216	0.6390

Using both word- and syllable-level indexing features  
& MCE Training