# Statistical Modeling Approaches for Information Retrieval

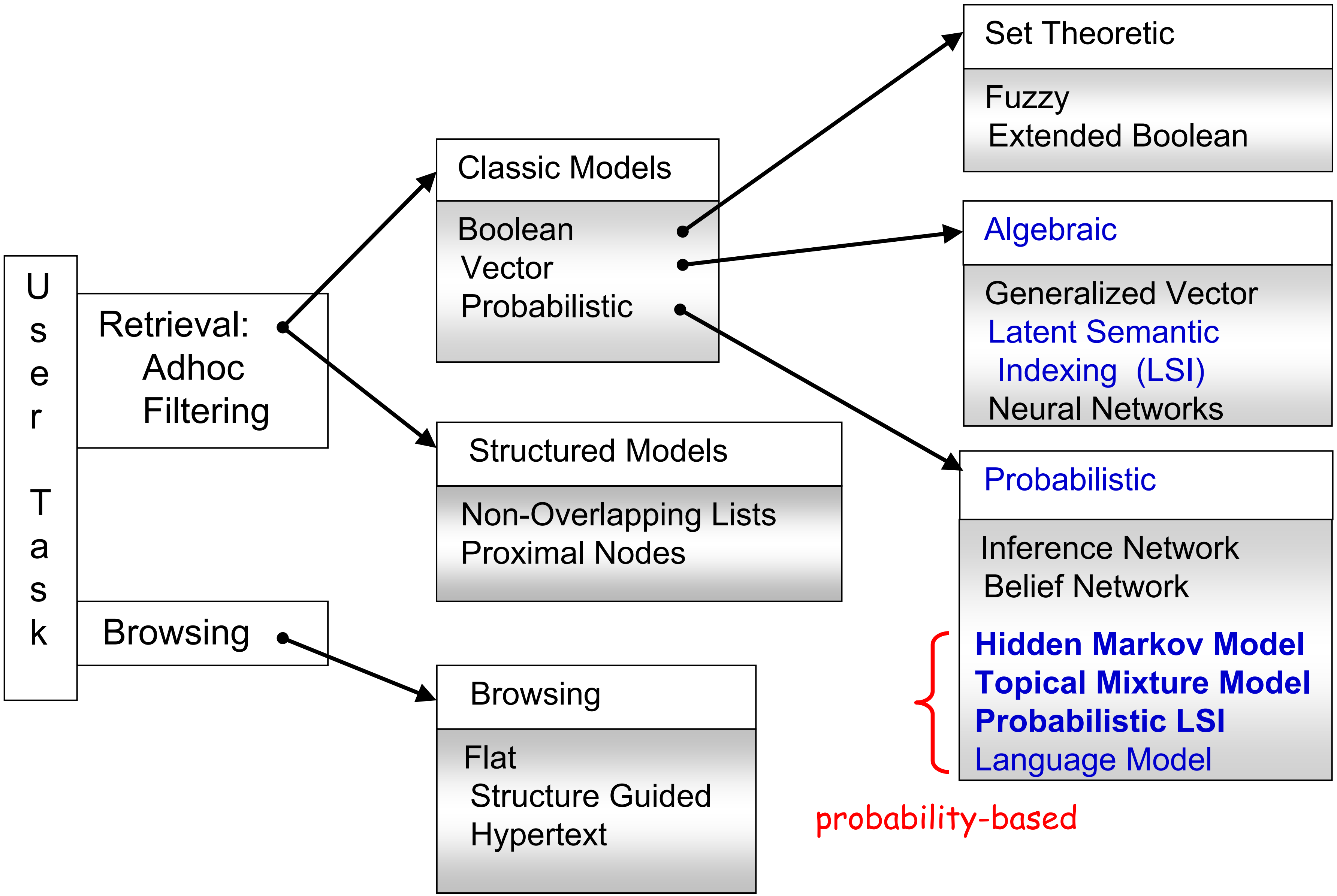
1. HMM/N-gram-based
2. Topical Mixture Model (TMM)
2. Latent Semantic Indexing (LSI)
3. Probabilistic Latent Semantic Analysis (PLSA)

Berlin Chen 2005

References:

1. W. B. Croft and J. Lafferty (Editors). *Language Modeling for Information Retrieval*. July 2003
2. B. 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. M. W. Berry et al. Using Linear Algebra for Intelligent Information Retrieval. Technical report, 1994
4. J.R. Bellegarda. *Latent semantic mapping*. IEEE Signal Processing Magazine, September 2005
5. T. Hofmann. *Unsupervised Learning by Probabilistic Latent Semantic Analysis*. Machine Learning, January-February 2001
6. B. Chen. *Exploring the Use of Latent Topical Information for Statistical Chinese Spoken Document Retrieval*, Pattern Recognition Letters 2005

# 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

# $N$-gram Language Model

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

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

  – How to estimate its corresponding probability ?

chain rule is applied

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

Too complicate to estimate all the necessary probability items !

# *N*-gram Language Model (cont.)

- *N*-gram approximation (Language Model, LM)
  - 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

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

  - ……..
- A variety of smoothing or interpolation techniques for *N*-gram LM have been proposed in the past several years

# 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 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, $Q = q_1 q_2 .. q_n .. q_N$, 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 (cont.)

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

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

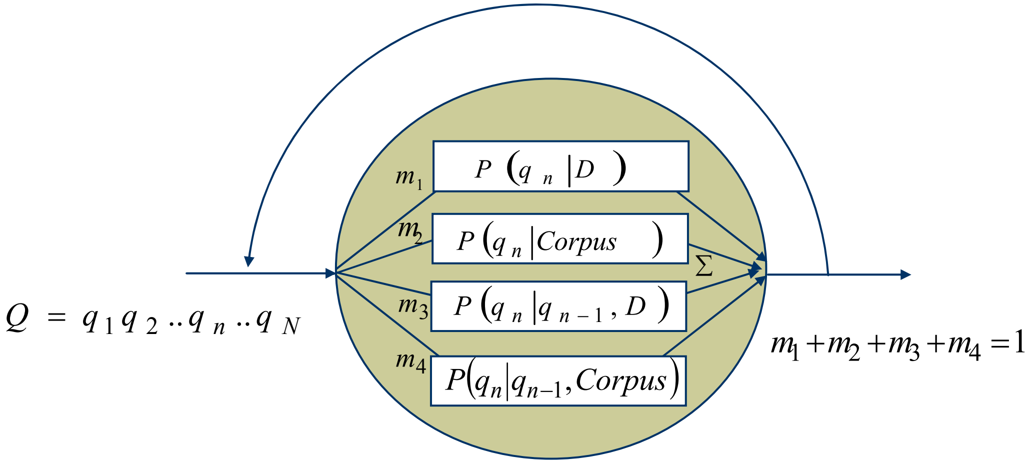
bigram modeling

**smoothing/interpolation**

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

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

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




*A mixture of N probability distributions*

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

- **Variants:** E.g., 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*)

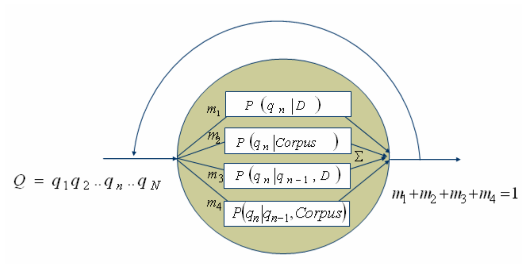
$$P(Q|D \text{ is } R) = \left[ m_1 P(q_1|D) + m_2 P(q_1|Corpus) \right] \cdot \prod_{n=2}^{N} \left[ m_1 P(q_n|D) + m_2 P(q_n|Corpus) + m_3 P(q_n|q_{n-1}, D) + m_4 P(q_n|q_{n-1}, Corpus) \right]$$

*P*(陳水扁 總統 視察 阿里山 小火車|*D*)
=[$m_1$*P*(陳水扁|*D*)+$m_2$ *P*(陳水扁|*C*)] x [$m_1$*P*(總統|*D*)+$m_2$*P*(總統|*C*)+ $m_3$*P*(總統|陳水扁, *D*)+$m_4$*P*(總統|陳水扁, *C*)]
x[$m_1$*P*(視察|*D*)+$m_2$*P*(視察|*C*)+ $m_3$*P*(視察|總統, *D*)+$m_4$*P*(視察|總統, *C*)] x .........

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

- Why Called "HMM"
  - The corresponding mixture sequence of *N*-gram components that generates the given observation sequence (the query) cannot be explicitly observed (or is non-deterministic)
  - E.g., $P(Q|D \text{ is } R)=\prod_{n=1}^{N}\left[m_1 P(q_n|D)+m_2 P(q_n|Corpus)\right]$ and $Q=q_1 q_2 q_3$
    - The possible mixture sequences *MIX* generate $Q$ would be

$$MIX_1 = mix_D, mix_D, mix_D$$
$$MIX_2 = mix_{Corpus}, mix_D, mix_D$$
$$MIX_3 = mix_D, mix_{Corpus}, mix_D$$
$$\vdots$$
$$MIX_8 = mix_{Corpus}, mix_{Corpus}, mix_{Corpus}$$




- "Hidden": The corresponding mixture sequence that generates the query can not be explicitly observed

# 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?)

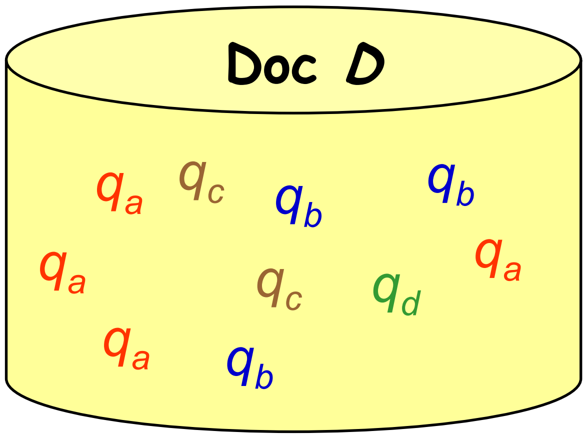
$P(q_n|Corpus)$

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

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

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

  - 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 (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*

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

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

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

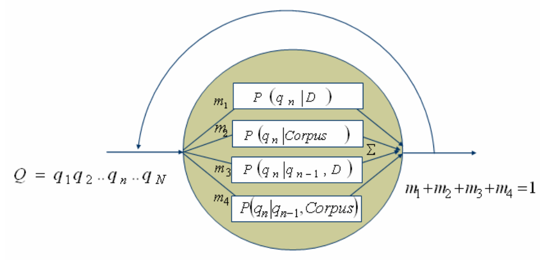
- Basically, $m_1$, $m_2$, $m_3$, $m_4$, can be estimated by using the Expectation-Maximization (EM) algorithm

  because of the insufficiency of training data

  - All docs share the same weights $m_i$ 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

# 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:

the old weight

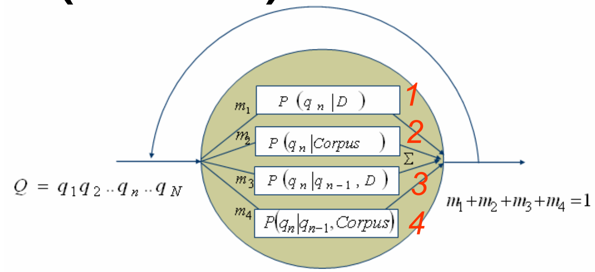
819 queries ≦2265 docs

the new weight

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

- 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 (cont.)




- Expectation-Maximization Training
  - Step 1: Expectation

the model

$$P\left(\boldsymbol{Q}, \boldsymbol{K} \mid \hat{D}\right) = P\left(\boldsymbol{K} \mid \boldsymbol{Q}, \hat{D}\right) P\left(\boldsymbol{Q} \mid \hat{D}\right)$$

query word sequence

mixture sequence

$$\boldsymbol{Q} = q_1 q_2 \cdots q_{N-1} q_N$$
$$\boldsymbol{K} = k_1 k_2 \cdots k_{N-1} k_N$$

- Log-likelihood expression and take expectation over ***K***

Take expectation on all possible mixture sequences K (conditioned on Q,D )

$$\log P\left(\boldsymbol{Q}, \boldsymbol{K} \mid \hat{D}\right) = \log P\left(\boldsymbol{K} \mid \boldsymbol{Q}, \hat{D}\right) + \log P\left(\boldsymbol{Q} \mid \hat{D}\right)$$

$$\Rightarrow \log P\left(\boldsymbol{Q} \mid \hat{D}\right) = \log P\left(\boldsymbol{Q}, \boldsymbol{K} \mid \hat{D}\right) - \log P\left(\boldsymbol{K} \mid \boldsymbol{Q}, \hat{D}\right)$$

$$\Rightarrow \mathrm{E}\left[\log P\left(\boldsymbol{Q} \mid \hat{D}\right)\right]_{\boldsymbol{K}|\boldsymbol{Q},D} = \mathrm{E}\left[\log P\left(\boldsymbol{Q}, \boldsymbol{K} \mid \hat{D}\right) - \log P\left(\boldsymbol{K} \mid \boldsymbol{Q}, \hat{D}\right)\right]_{\boldsymbol{K}|\boldsymbol{Q},D}$$

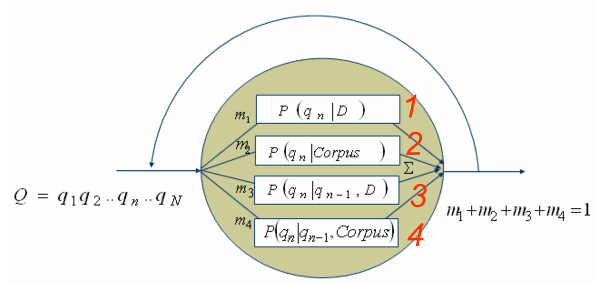
$$\Rightarrow \sum_{\boldsymbol{K}} P(\boldsymbol{K}|\boldsymbol{Q},D) \log P\left(\boldsymbol{Q} \mid \hat{D}\right) = \sum_{\boldsymbol{K}} P(\boldsymbol{K}|\boldsymbol{Q},D)\left(\log P\left(\boldsymbol{Q}, \boldsymbol{K} \mid \hat{D}\right) - \log P\left(\boldsymbol{K} \mid \boldsymbol{Q}, \hat{D}\right)\right)$$

$$\Rightarrow \log P\left(\boldsymbol{Q} \mid \hat{D}\right) = \sum_{\boldsymbol{K}} P(\boldsymbol{K}|\boldsymbol{Q},D) \log P\left(\boldsymbol{Q}, \boldsymbol{K} \mid \hat{D}\right) - \sum_{\boldsymbol{K}} P(\boldsymbol{K}|\boldsymbol{Q},D) \log P\left(\boldsymbol{K} \mid \boldsymbol{Q}, \hat{D}\right)$$

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

- **Explanation**

$M$ mixtures of distributions




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

$$=\left(m_1P(q_1|k_1,\hat{D})+\cdots+m_MP(q_1|k_M,\hat{D})\right)\times\left(m_1P(q_2|k_1,\hat{D})+\cdots+m_MP(q_2|k_M,\hat{D})\right)$$
$$\times\cdots\times\left(m_1P(q_N|k_1,\hat{D})+\cdots+m_MP(q_N|k_M,\hat{D})\right)$$

Note

$$\prod_{t=1}^{T}\left(\sum_{k_t=1}^{M}a_{tk_t}\right)$$
$$=(a_{11}+a_{12}+\ldots+a_{1M})(a_{21}+a_{22}+\ldots+a_{2M})\ldots(a_{T1}+a_{T2}+\ldots+a_{TM})$$
$$=\sum_{k_1=1}^{M}\sum_{k_2=1}^{M}\ldots\sum_{k_T=1}^{M}\prod_{t=1}^{T}a_{tk_t}$$

Sum-product → product-sum

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

where $m_{k_n}=P(k_n|\hat{D})$

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

$\boldsymbol{Q}=q_1q_2\cdots q_{N-1}q_N$

$\boldsymbol{K}=k_1k_2\cdots k_{N-1}k_N$

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

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

Independence Assumption

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

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

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

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

- Expectation-Maximization Training
  - Step 1: Expectation (cont.)
    - Express $\log P\left(\boldsymbol{Q} \mid \hat{D}\right)$ using two auxiliary functions

$$\log P\left(\boldsymbol{Q} \mid \hat{D}\right) = \sum_{\boldsymbol{K}} P\left(\boldsymbol{K} \middle| \boldsymbol{Q}, D\right) \log P\left(\boldsymbol{Q}, \boldsymbol{K} \mid \hat{D}\right) - \sum_{\boldsymbol{K}} P\left(\boldsymbol{K} \middle| \boldsymbol{Q}, D\right) \log P\left(\boldsymbol{K} \mid \boldsymbol{Q}, \hat{D}\right)$$

where

$$\log P\left(\boldsymbol{Q} \mid \hat{D}\right) = \Phi\left(D, \hat{D}\right) - \mathrm{H}\left(D, \hat{D}\right)$$

complete data

$$\Phi\left(D, \hat{D}\right) = E\left[L^{C}\right] = \sum_{\boldsymbol{K}} P\left(\boldsymbol{K} \middle| \boldsymbol{Q}, D\right) \log P\left(\boldsymbol{Q}, \boldsymbol{K} \mid \hat{D}\right)$$

$$\mathrm{H}\left(D, \hat{D}\right) = \sum_{\boldsymbol{K}} P\left(\boldsymbol{K} \middle| \boldsymbol{Q}, D\right) \log P\left(\boldsymbol{K} \mid \boldsymbol{Q}, \hat{D}\right)$$

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

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

unknown model setting

$$\log P\left(\boldsymbol{Q} \mid \hat{D}\right) \geq \log P\left(\boldsymbol{Q} \mid D\right)$$

$$\begin{aligned}
&\log P\left(\boldsymbol{Q} \mid \hat{D}\right) - \log P\left(\boldsymbol{Q} \mid D\right) \\
&= \left[\Phi\left(D, \hat{D}\right) - \mathrm{H}\left(D, \hat{D}\right)\right] - \left[\Phi\left(D, D\right) - \mathrm{H}\left(D, D\right)\right] \\
&= \Phi\left(D, \hat{D}\right) - \Phi\left(D, D\right) \boxed{- \mathrm{H}\left(D, \hat{D}\right) + \mathrm{H}\left(D, D\right)}
\end{aligned}$$

$$\geq 0$$

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

- Expectation-Maximization Training
  - Step 1: Expectation (cont.)
    - $-\mathrm{H}\left(D,\hat{D}\right)+\mathrm{H}(D,D)$ has the following property

$$
\begin{aligned}
&-\mathrm{H}\left(D,\hat{D}\right)+\mathrm{H}(D,D) \\
&= -\left[\sum_{\boldsymbol{K}} P(\boldsymbol{K}|\boldsymbol{Q},D)\log P\left(\boldsymbol{K}\middle|\boldsymbol{Q},\hat{D}\right)\right] + \left[\sum_{\boldsymbol{K}} P(\boldsymbol{K}|\boldsymbol{Q},D)\log P(\boldsymbol{K}|\boldsymbol{Q},D)\right] \\
&= -\sum_{\boldsymbol{K}}\left[P(\boldsymbol{K}|\boldsymbol{Q},D)\log\frac{P\left(\boldsymbol{K}\middle|\boldsymbol{Q},\hat{D}\right)}{P(\boldsymbol{K}|\boldsymbol{Q},D)}\right] \\
&\geq \sum_{\boldsymbol{K}}\left[P(\boldsymbol{K}|\boldsymbol{Q},D)\left(1-\frac{P\left(\boldsymbol{K}\middle|\boldsymbol{Q},\hat{D}\right)}{P(\boldsymbol{K}|\boldsymbol{Q},D)}\right)\right] \\
&= -\sum_{\boldsymbol{K}}\left[P(\boldsymbol{K}|\boldsymbol{Q},D)-P\left(\boldsymbol{K}\middle|\boldsymbol{Q},\hat{D}\right)\right] \\
&= 0
\end{aligned}
$$

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

*Kullbuack-Leibler (KL) distance*

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

*Jensen's inequality*

$(-\log x \geq 1-x)$

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

- Expectation-Maximization Training
  - Step 1: Expectation (cont.)
    - Therefore, for maximizing $\log P\left(\boldsymbol{Q} \mid \hat{D}\right)$ , we only need to maximize the $\Phi$-function (auxiliary function)

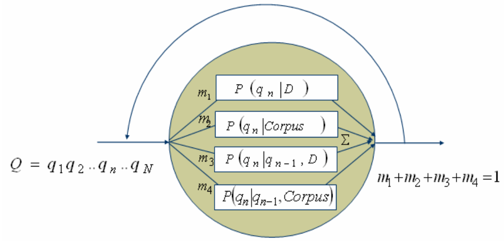
$$\Phi\left(D, \hat{D}\right) = \sum_{\boldsymbol{K}} P\left(\boldsymbol{K} \middle| \boldsymbol{Q}, D\right) \log P\left(\boldsymbol{Q}, \boldsymbol{K} \mid \hat{D}\right)$$

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

$$\Phi\left(D, \hat{D}\right) = \sum_{\boldsymbol{K}} P\left(\boldsymbol{K} \middle| \boldsymbol{Q}, D\right) \log P\left(\boldsymbol{Q}, \boldsymbol{K} \mid \hat{D}\right)$$
$$\overset{?}{=} \sum_{q_n \in \boldsymbol{Q}} \sum_{k} P\left(k \middle| q_n, D\right) \log P\left(q_n, k \mid \hat{D}\right)$$

?

# 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

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

Auxiliary function

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

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

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

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

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

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

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

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

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

By applying Lagrange Multiplier $\ell$

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

**Constraint**

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

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

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

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

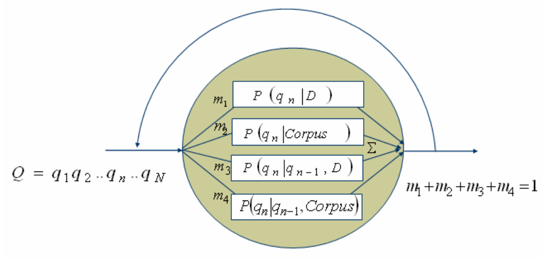
$$\overline{\Phi}_{\boldsymbol{m}} = \sum_{q_n \in \boldsymbol{Q}} \sum_k \left\{ \frac{P(q_n|k,D)\, m_k}{\sum_j P(q_n|j,D)\, m_j} \log \hat{m}_k \right\} + \boxed{l\left(\sum_i \hat{m}_i - 1\right)}$$

normalization constraints using Lagrange multipliers

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

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

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




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

$$\because \quad l = -\sum_s G_s$$

$$\therefore \quad \hat{m}_k = \frac{\overbrace{\sum_{q_n \in Q} \frac{P(q_n|k,D)\,m_k}{\sum_j P(q_n|j,D)\,m_j}}^{G_k}}{\underbrace{\sum_s \sum_{q_n \in Q} \frac{P(q_n|k,D)\,m_s}{\sum_j P(q_n|j,D)\,m_j}}_{l}} = \frac{\sum_{q_n \in Q} \frac{P(q_n|k,D)\,m_k}{\sum_j P(q_n|j,D)\,m_j}}{|Q|}$$

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

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

# 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 | **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** |

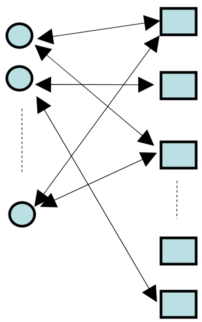
  - HMM/N-gram-based approach is consistently better than vector space model

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

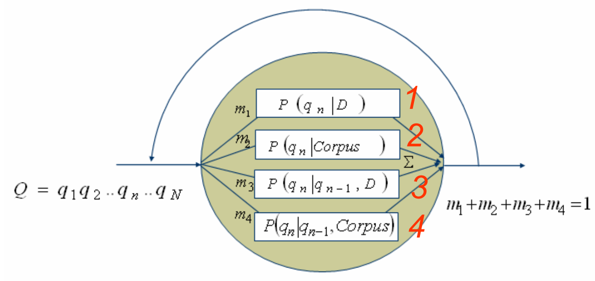
- Mixture Observation Probabilities also can be optimized by EM training

$$\hat{P}(q_n|k,D)=\frac{\sum_{Q\in[TrainSet]_Q}\ \sum_{\text{st. } D\in[Doc]_{R\text{ to }Q}}\ \sum_{q\in Q,\, q=q_n}\left[\frac{P(q|k,D)m_k}{\sum_{m_j}P(q|j,D)m_j}\right]}{\sum_{Q\in[TrainSet]_Q}\ \sum_{\text{st. } D\in[Doc]_{R\text{ to }Q}}\ \sum_{q\in Q}\left[\frac{P(q|k,D)m_k}{\sum_{m_j}P(q|j,D)m_j}\right]}$$

Training queries



Documents in the collection



# 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 conditioned on the current estimate of the parameters and the observations
    - ***M***: provides a new estimation of the parameters according to ML (or MAP)

Jeff A. Bilmes "A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models," U.C. Berkeley TR-97-021

# 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 $\log P(\boldsymbol{O}|\lambda)$ by iteratively maximizing the expectation of log-likelihood from complete data $\log P(\boldsymbol{O},\boldsymbol{S}|\lambda)$

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

$$\Phi\left(\lambda,\hat{\lambda}\right)=\sum_{\boldsymbol{O}} E\left[\log P\left(\boldsymbol{O},\boldsymbol{S}\middle|\hat{\lambda}\right)\right]_{\boldsymbol{S}|\boldsymbol{O},\lambda}$$

# 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:

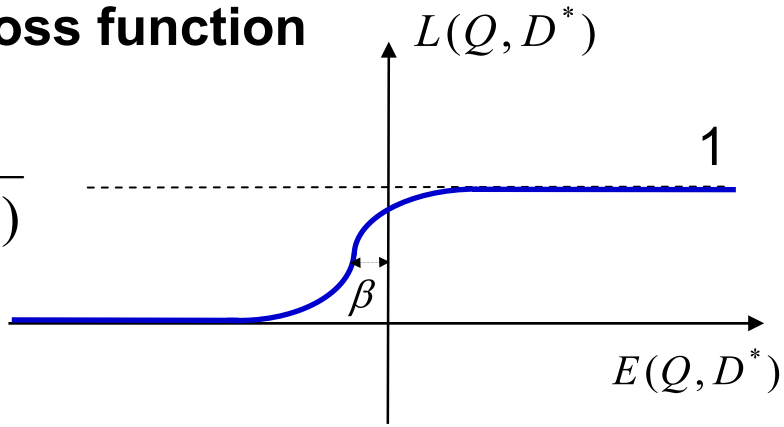
$$E(Q,D^*)=\frac{1}{|Q|}\left[-\log P\left(Q\middle|D^* \text{ is } R\right)+\max_{D'}\log P\left(Q\middle|D' \text{ is not } R\right)\right]$$

Also can take all irrelevant doc in the answer set into consideration

">0": means misclassified; "<=0": means a correct decision

  - Transform the error function to **the loss function**

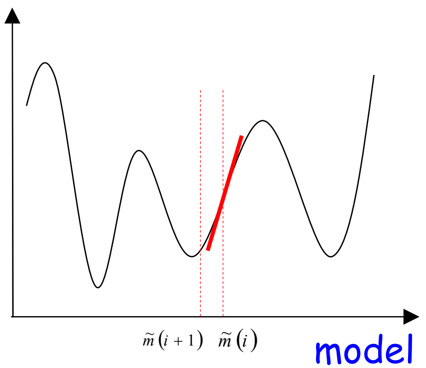
$$L(Q,D^*)=\frac{1}{1+\exp(-\alpha E(Q,D^*)+\beta)}$$




- In the range between 0 and 1
  - $\alpha$ : controls the slope
  - $\beta$ : controls the offset

# 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

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

    - Where,

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

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

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

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

Note :

$$[\log f(x)]' = \frac{1}{f(x)} f'(x)$$

$$[f(x)g(x)]' = f'(x)g(x) + f(x)g'(x)$$

$$\left[\frac{f(x)}{g(x)}\right]' = \frac{f'(x)g(x) - f(x)g'(x)}{g^2(x)}$$

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

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

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

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

# 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

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

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

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

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

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 (cont.)

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

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

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

- $m_2$ can be updated in the similar way

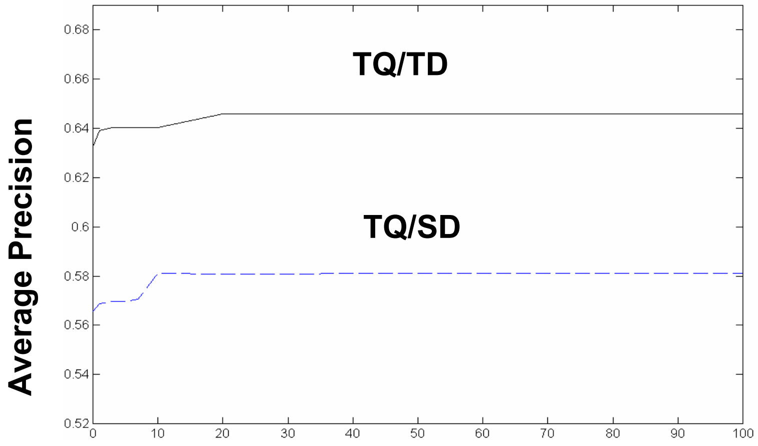
# HMM/N-gram-based Model (cont.)

- Experimental results with MCE training

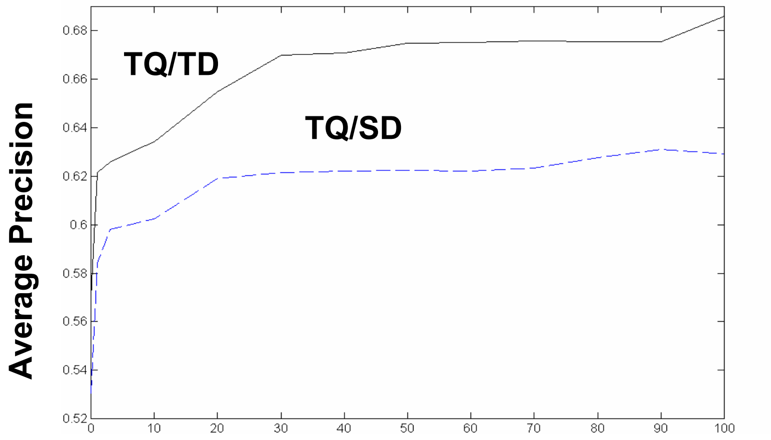
**Before MCE Training**

| Average Precision | | Word-level Uni | Syllable-level Uni+Bi* | Fusion |
|---|---|---|---|---|
| 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




MCE Iterations (Word-based)




MCE Iterations (syllable-based)

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

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

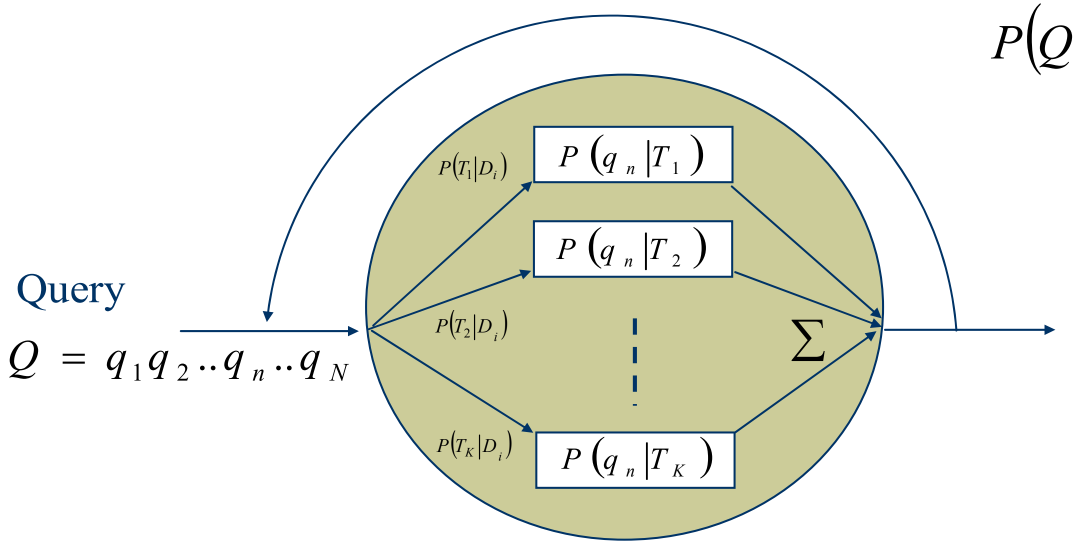
- Advantages
  - A formal mathematic framework
  - Use collection statistics but not heuristics
  - The retrieval system can be gradually improved through usage

- 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
- Like HMM, Various Theoretically Attractive Model Training Algorithms can be applied
  - Maximum likelihood (e.g. EM) or discriminative (e.g. MCE or MMI) training

A document model




$$P(Q|D_i) \approx \prod_{n=1}^{N}\sum_{k=1}^{K} P(q_n|T_k)P(T_k|D_i)$$

- $P(q_n|T_k)$ are tied among docs
- Each doc has its own $P(T_k|D_i)$

# Topical Mixture Model (cont.)

- EM Training (supervision)
  - Given a training set of query exemplars with the corresponding query-document relevance information

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

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

$$|Q| = \sum_{q_s \in Q} n(q_s, Q)$$

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

# Topical Mixture Model (cont.)

- EM Training ( without supervision)
  - Use each document itself as a a query exemplar to train its own document mixture model

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

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

$$\left|D_i\right| = \sum_{w_s \in D_i} n\left(w_s, D_i\right)$$

, where $\quad P\left(T_k \middle| q_n, D_i\right) = \dfrac{P\left(T_k \middle| D_i\right) P\left(w_n \middle| T_k\right)}{\sum_{l=1}^{K} P\left(T_l \middle| D_i\right) P\left(w_n \middle| T_l\right)}$

# Topical Mixture Model (cont.)

- Probability Smoothing
  - With corpus unigram probability $P_{ML}\left(q_n \middle| Corpus\right)$

$$\hat{P}\left(Q \middle| D_i\right) \approx \prod_{n=1}^{N}\left[\alpha\left(\sum_{k=1}^{K} P\left(q_n \middle| T_k\right) P\left(T_k \middle| D_i\right)\right) + (1-\alpha)\underline{P_{ML}\left(q_n \middle| Corpus\right)}\right]$$

  - Additionally with doc unigram probability $P_{ML}\left(q_n \middle| D_i\right)$

$$\hat{P}\left(Q \middle| D_i\right) \approx \prod_{n=1}^{N}\left\{\beta\left[\alpha\left(\sum_{k=1}^{K} P\left(q_n \middle| T_k\right) P\left(T_k \middle| D_i\right)\right) + (1-\alpha) P_{ML}\left(q_n \middle| Corpus\right)\right] + (1-\beta)\underline{P_{ML}\left(q_n \middle| D_i\right)}\right\}$$

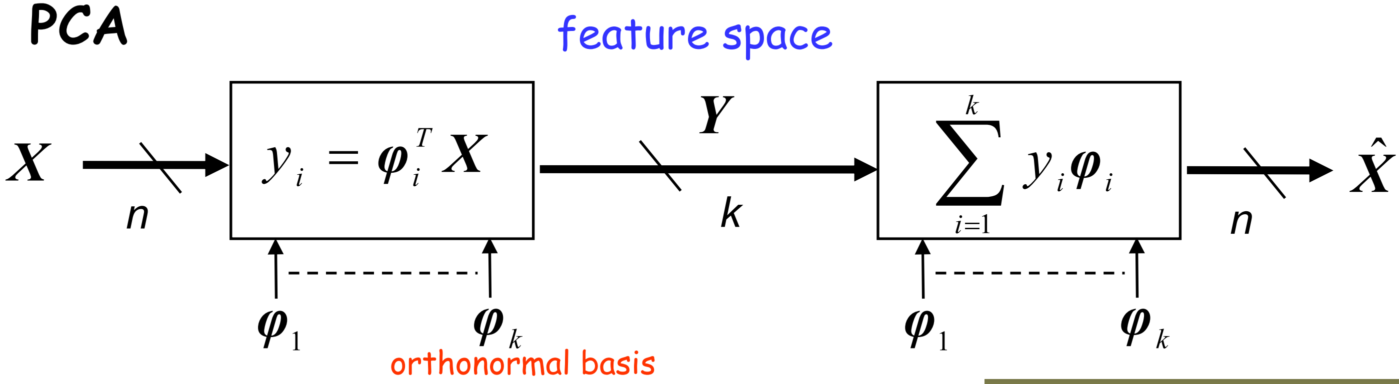
$$Q = q_1 q_2 \ldots q_n \ldots q_N$$

# Latent Semantic Indexing (LSI)

- Also called Latent Semantic Analysis (LSA), Latent Semantic Mapping (LSM)
- 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)

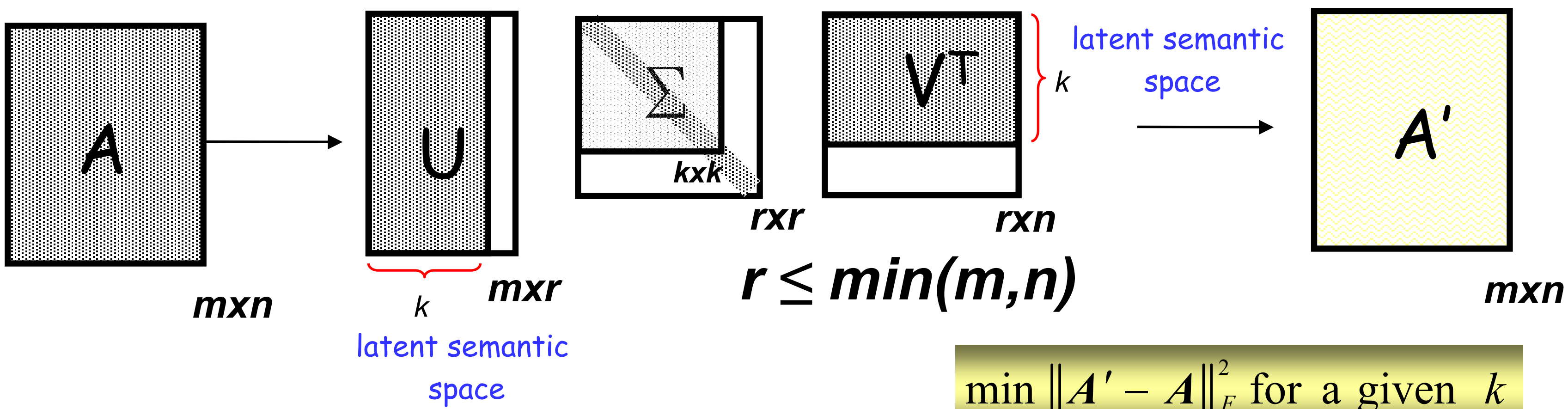
# Latent Semantic Indexing (cont.)

- Dimension Reduction and Feature Extraction
  - PCA




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

  - SVD (in LSI)



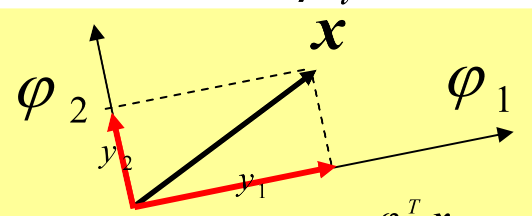

$$\min \left\| A' - A \right\|_F^2 \quad \text{for a given } k$$

# Latent Semantic Indexing (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 $\boldsymbol{x}$ : $\boldsymbol{\varphi}_i$

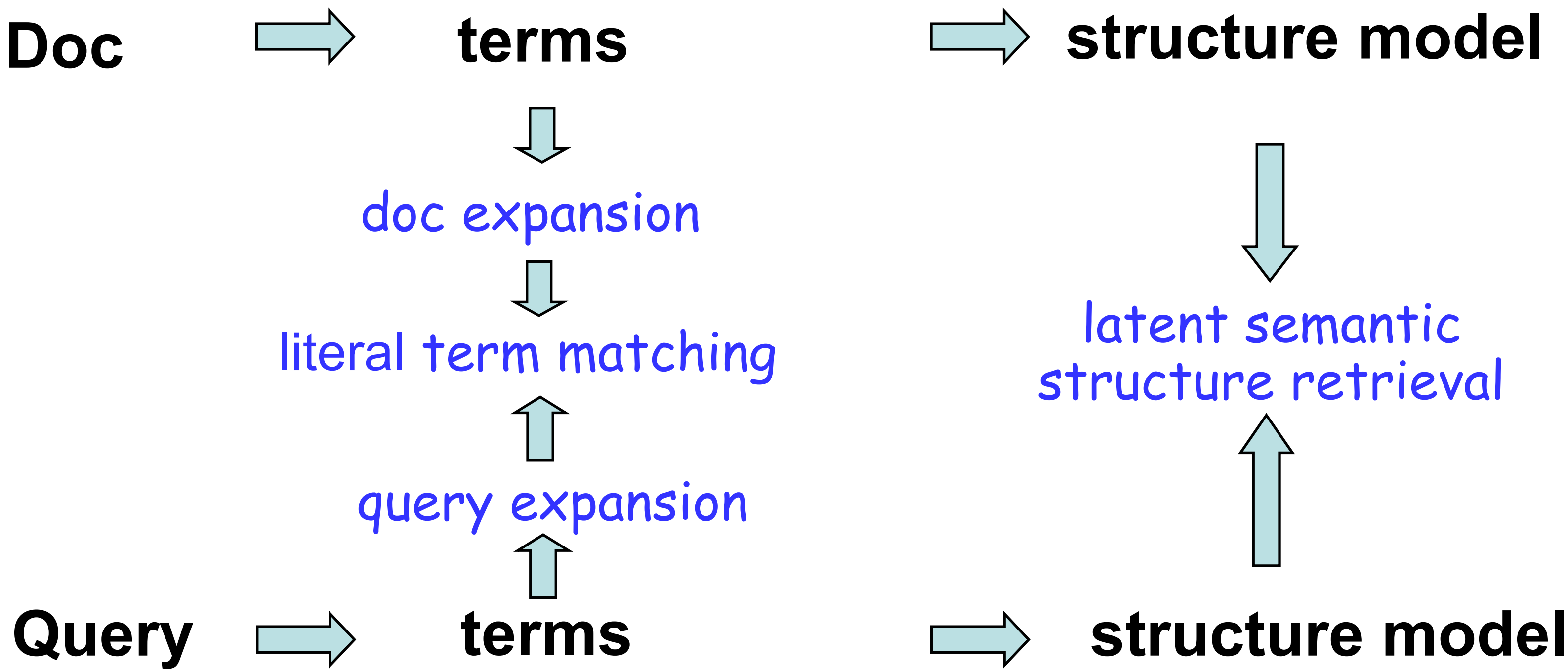



$$y_1 = \|\boldsymbol{x}\| \cos\theta_1 = \|\boldsymbol{x}\| \frac{\varphi_1^T \boldsymbol{x}}{\|\boldsymbol{x}\| \|\boldsymbol{\varphi}_1\|} = \varphi_1^T \boldsymbol{x}$$

, where $\|\boldsymbol{\varphi}_1\| = 1$

# 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 |
| *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 |

# Latent Semantic Indexing (cont.)

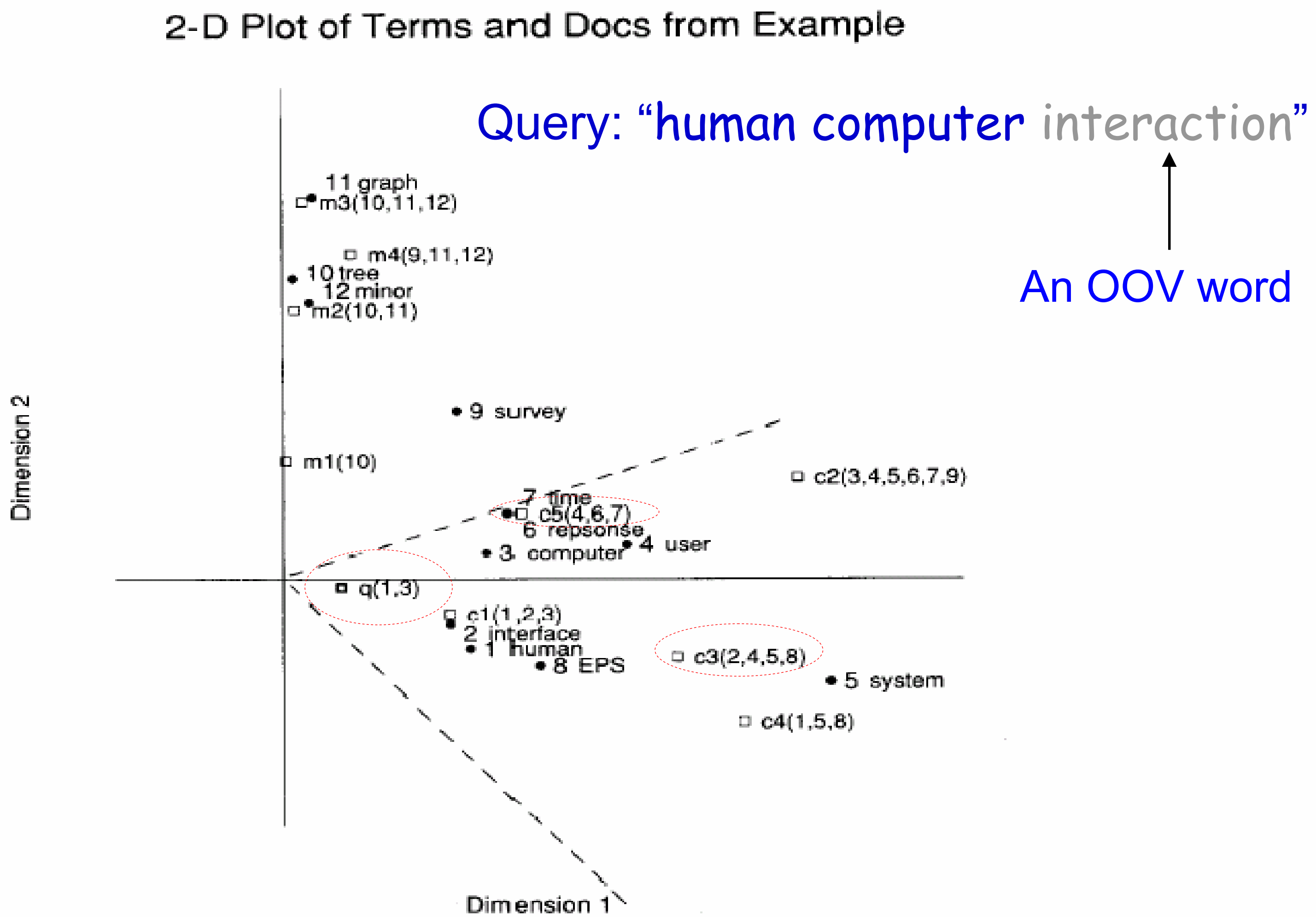


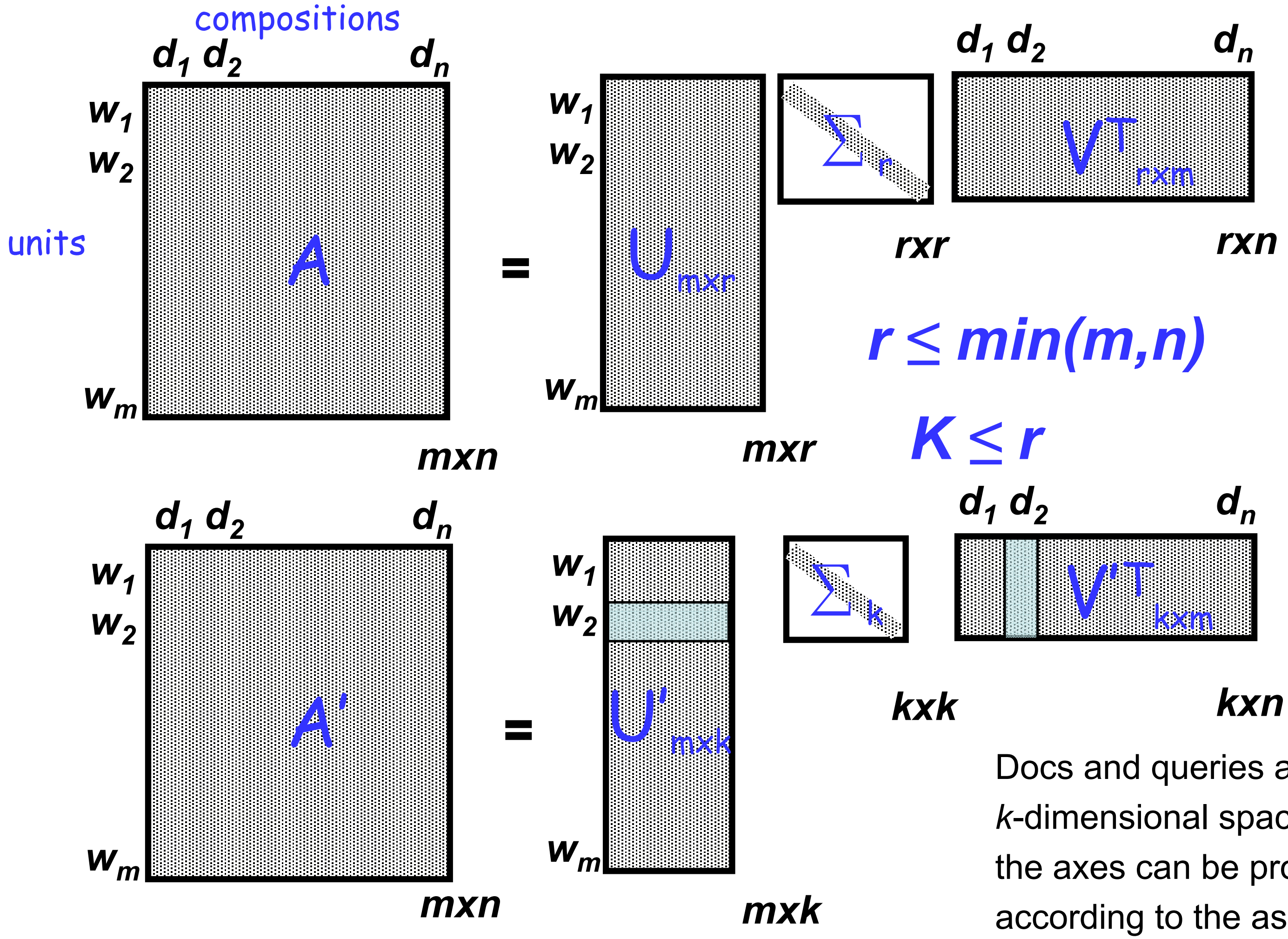

FIG. 1. A two-dimensional plot of 12 Terms and 9 Documents from the sampe TM set. Terms are represented by filled circles. Documents are shown as open squares, and component terms are indicated parenthetically. The query ("human computer interaction") is represented as a pseudo-document at point $q$.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

- Singular Value Decomposition (SVD)

Row A $\in R^n$

Col A $\in R^m$




Both U and V has orthonormal column vectors

$U^TU=I_{rXr}$

$V^TV=I_{rXr}$

$r \leq min(m,n)$

$K \leq r$

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

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

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.)

- "term-document" matrix *A* has to do with the co-occurrences between terms (or units) and documents (or compositions)
  - Contextual information for words in documents is discarded
    - "bag-of-words" modeling

- **Feature extraction** for the entities $a_{i,j}$ of matrix *A*

  1. Conventional *tf-idf* statistics

  2. Or, $a_{i,j}$ :occurrence frequency weighted by negative entropy

occurrence count

$$a_{i,j} = \frac{f_{i,j}}{\left|d_j\right|} \times (1 - \varepsilon_i), \quad \left|d_j\right| = \sum_{i=1}^{m} f_{i,j}$$

negative normalized entropy

document length

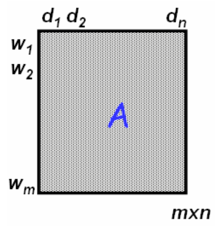
normalized entropy of term *i*

occurrence count of term *i* in the collection

$$\varepsilon_i = -\frac{1}{\log n} \sum_{j=1}^{n} \left( \frac{f_{i,j}}{\tau_i} \log \frac{f_{i,j}}{\tau_i} \right), \quad \tau_i = \sum_{j=1}^{n} f_{i,j}$$

$$0 \le \varepsilon_i \le 1$$

# Latent Semantic Indexing (cont.)




- Singular Value Decomposition (SVD)
  - $A^T A$ is symmetric $n$x$n$ matrix
    - All eigenvalues $\lambda_j$ 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 ( $\in R^n$)

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

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

$$\sigma_1 = \|Av_1\|$$
$$\sigma_2 = \|Av_2\|$$
.....

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

# Latent Semantic Indexing (cont.)

- $\{Av_1, Av_2, \ldots, Av_r\}$ 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 $A^TA$) has rank $r \le n$

$$\lambda_1 \ge \lambda_2 \ge \ldots. \ge \lambda_r > 0, \quad \lambda_{r+1} = \lambda_{r+2} = \ldots. = \lambda_n = 0$$

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

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

*u*: also an orthonormal matrix (mxr)

*V*: an orthonormal matrix

$$\Rightarrow [u_1\ u_2 \ldots u_r]\Sigma_r = A\boxed{[v_1\ v_2\ \ \ v_r]}$$

Known in advance

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

$$\Rightarrow [u_1\ u_2 \ldots u_r \ldots u_m]\Sigma = A[v_1\ v_2 \ldots v_r \ldots v_n]$$

$$\Rightarrow U\Sigma = AV \quad \Rightarrow U\Sigma V^T = A\underbrace{VV^T}_{I_{nxn}}$$

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

$$\Sigma_{m\times n} = \begin{pmatrix} \Sigma_r & \mathbf{0}_{r\times(n-r)} \\ \mathbf{0}_{(m-r)\times r} & \mathbf{0}_{(m-r)\times(n-r)} \end{pmatrix}$$

?

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

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

# Latent Semantic Indexing (cont.)

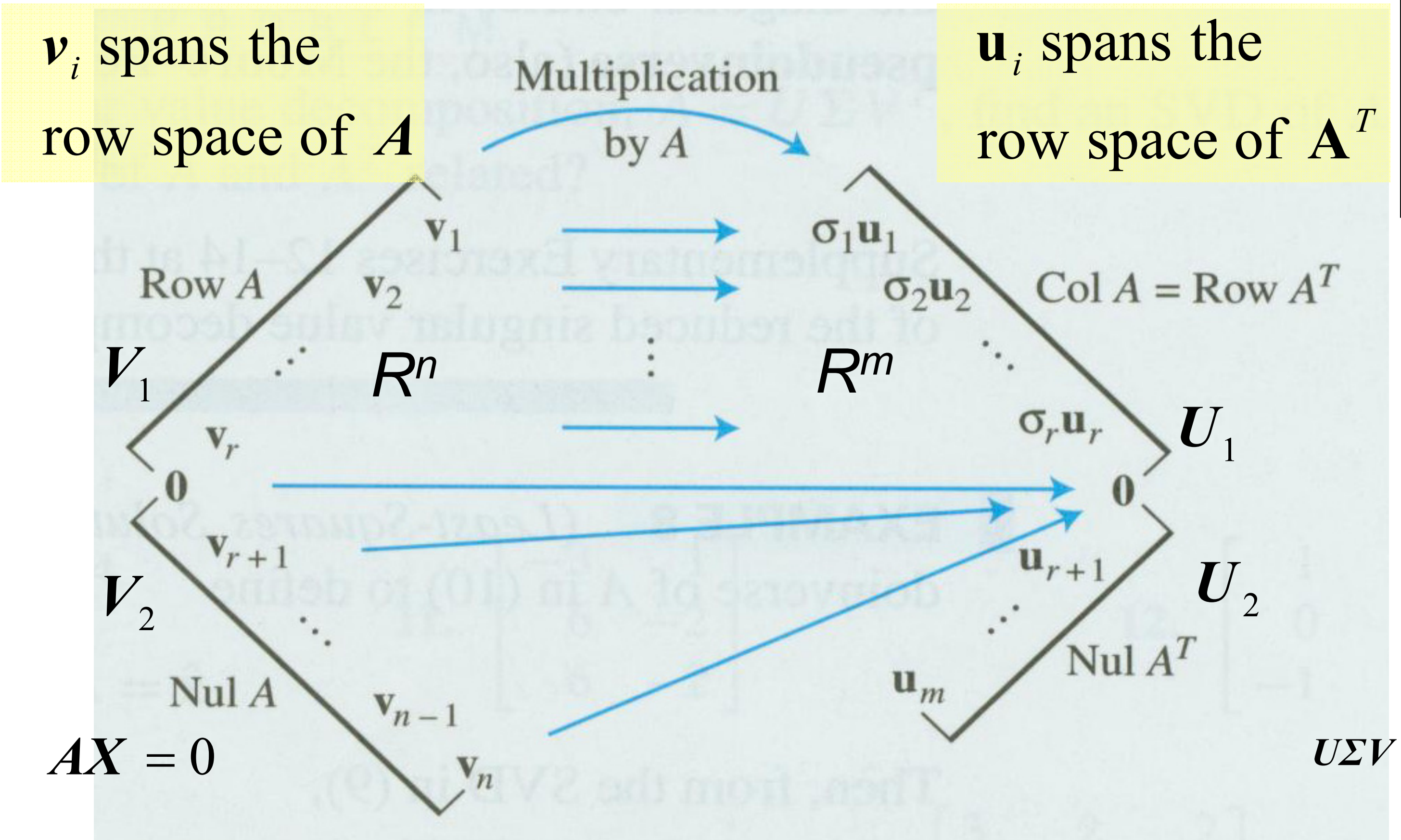
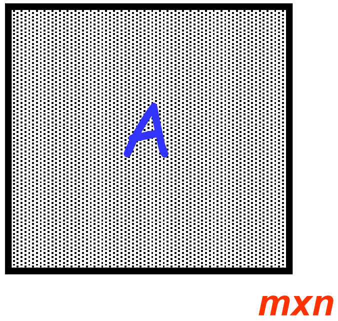
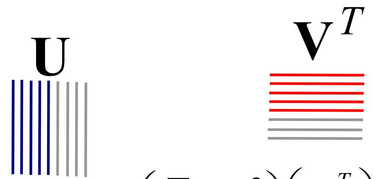



**FIGURE 4** The four fundamental subspaces and the action of $A$.




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

$U\Sigma = AV$

# Latent Semantic Indexing (cont.)

- Additional Explanations
  - Each row of $U$ is related to the projection of a corresponding row of $A$ onto the basis formed by columns of $V$

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

$$\Rightarrow AV = U\Sigma V^T V = U\Sigma \quad \Rightarrow \quad U\Sigma = AV$$

    - the *i*-th entry of a row of $U$ is related to the projection of a corresponding row of $A$ onto the *i*-th column of $V$

  - Each row of $V$ is related to the projection of a corresponding row of $A^T$ onto the basis formed by $U$

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

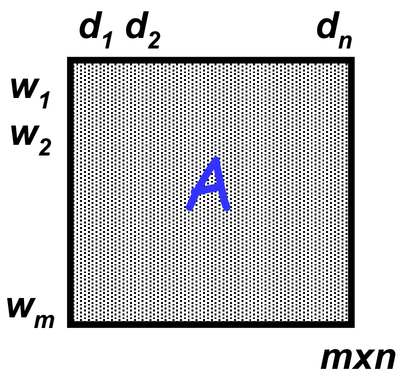
$$\Rightarrow A^T U = \left(U\Sigma V^T\right)^T U = V\Sigma U^T U = V\Sigma$$

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

    - the *i*-th entry of a row of $V$ is related to the projection of a corresponding row of $A^T$ onto the *i*-th column of $U$

# Latent Semantic Indexing (cont.)

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



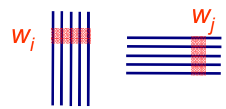

- compare two terms $\rightarrow$ dot product of two rows of A
  - or an entry in $AA^T$
- compare two docs $\rightarrow$ dot product of two columns of A
  - or an entry in $A^TA$
- compare a term and a doc $\rightarrow$ each individual entry of A

  - The new word-document matrix (A')

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

$\Sigma'=\Sigma_k$

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

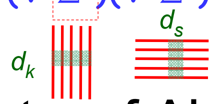



- compare two terms $\rightarrow$ dot product of two rows of $U'\Sigma'$

$$A'A'^T=(U'\Sigma'V'^T)(U'\Sigma'V'^T)^T=U'\Sigma'\underbrace{\boxed{V'^TV'}}_{I_{r\times r}}\Sigma'^TU'^T=(U'\underbrace{\boxed{\Sigma'}}_{\text{For stretching or shrinking}})(U'\Sigma')^T$$

- compare two docs $\rightarrow$ dot product of two rows of $V'\Sigma'$

$$A'^TA'=(U'\Sigma'V'^T)^T(U'\Sigma'V'^T)=V'\Sigma'^T\underbrace{\boxed{U^TU'}}_{I_{r\times r}}\Sigma'V'^T=(V'\underbrace{\boxed{\Sigma'}}_{\text{For stretching or shrinking}})(V'\Sigma')^T$$




- compare a query word and a doc $\rightarrow$ each individual entry of A'

# Latent Semantic Indexing (cont.)

- **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

See Figure A in next page

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

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

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

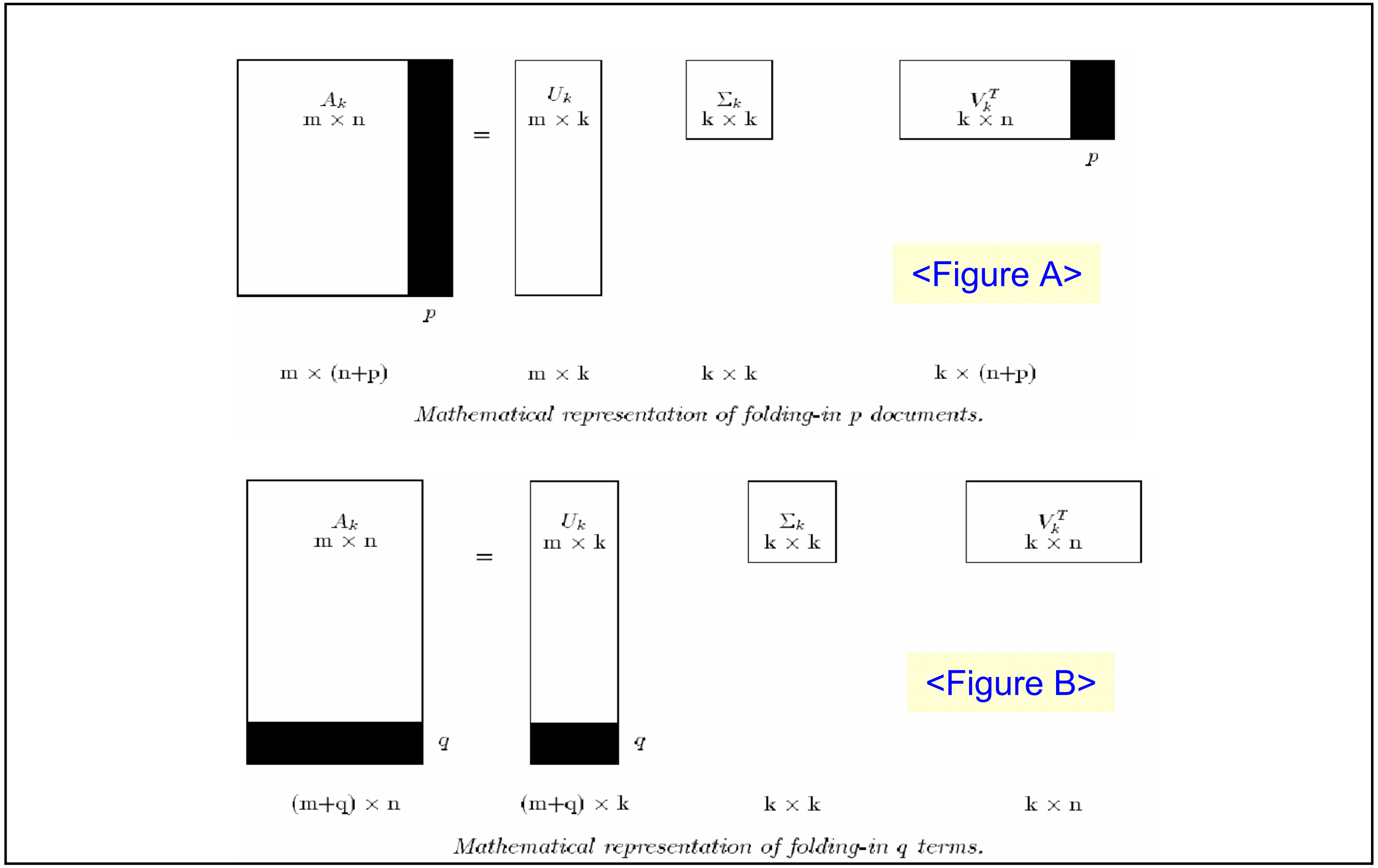
row vectors

# Latent Semantic Indexing (cont.)

- Fold-in a new 1 x n term vector

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

See Figure B below

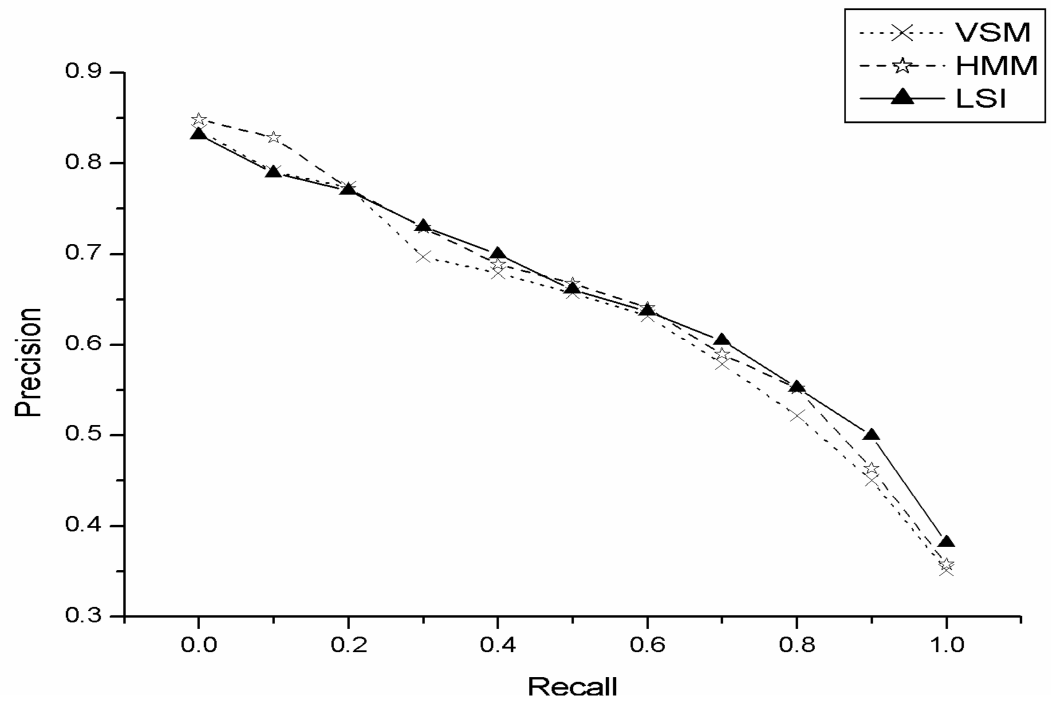



*Mathematical representation of folding-in p documents.*

*Mathematical representation of folding-in q terms.*

# 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 (a kind of Bayesian Networks)

$$\begin{CD} @>{P(D_i)}>> D @>{P(w_n|D_i)}>> Q \end{CD}$$

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

$$\begin{CD} @>{P(D_i)}>> D @>{P(T_k|D_i)}>> T @>{P(w_n|T_k)}>> Q \end{CD}$$

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

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

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

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

?

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

# 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

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

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

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

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

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

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

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

take expectation

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

complete data likelihood

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

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

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

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 $\hat{P}\left(T_k \middle| w_j, D_i\right)$ can be further decomposed as

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

    - The auxiliary function

Kullback-Leibler divergence

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

# Probabilistic Latent Semantic Analysis (cont.)

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

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

normalization constraints using Lagrange multipliers

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

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

# Probabilistic Latent Semantic Analysis (cont.)

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

The training formula

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

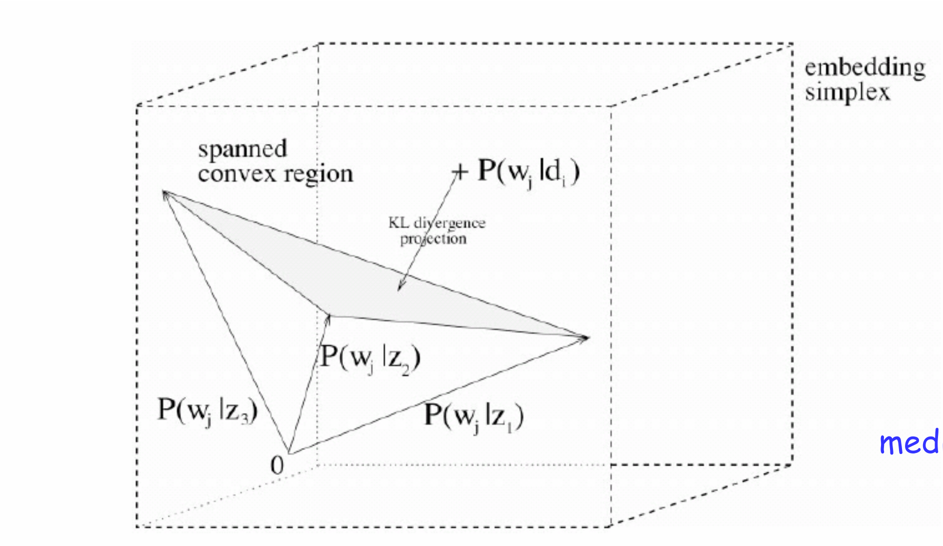
$$\hat{P}\left(T_k \mid D_i\right)=\frac{\sum_{w_j} n\left(w_j, D_i\right) P\left(T_k \mid w_j, D_i\right)}{\sum_{T_k} \sum_{w_j} n\left(w_j, D_i\right) P\left(T_k \mid w_j, D_i\right)}=\frac{\sum_{w_j} n\left(w_j, D_i\right) P\left(T_k \mid w_j, D_i\right)}{\sum_{w_j} n\left(w_j, D_i\right)}$$

$$=\boxed{\frac{\sum_{w_j} n\left(w_j, D_i\right) P\left(T_k \mid w_j, D_i\right)}{n\left(D_i\right)}}$$

The training formula

# Probabilistic Latent Semantic Analysis (cont.)

- Latent Probability Space




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

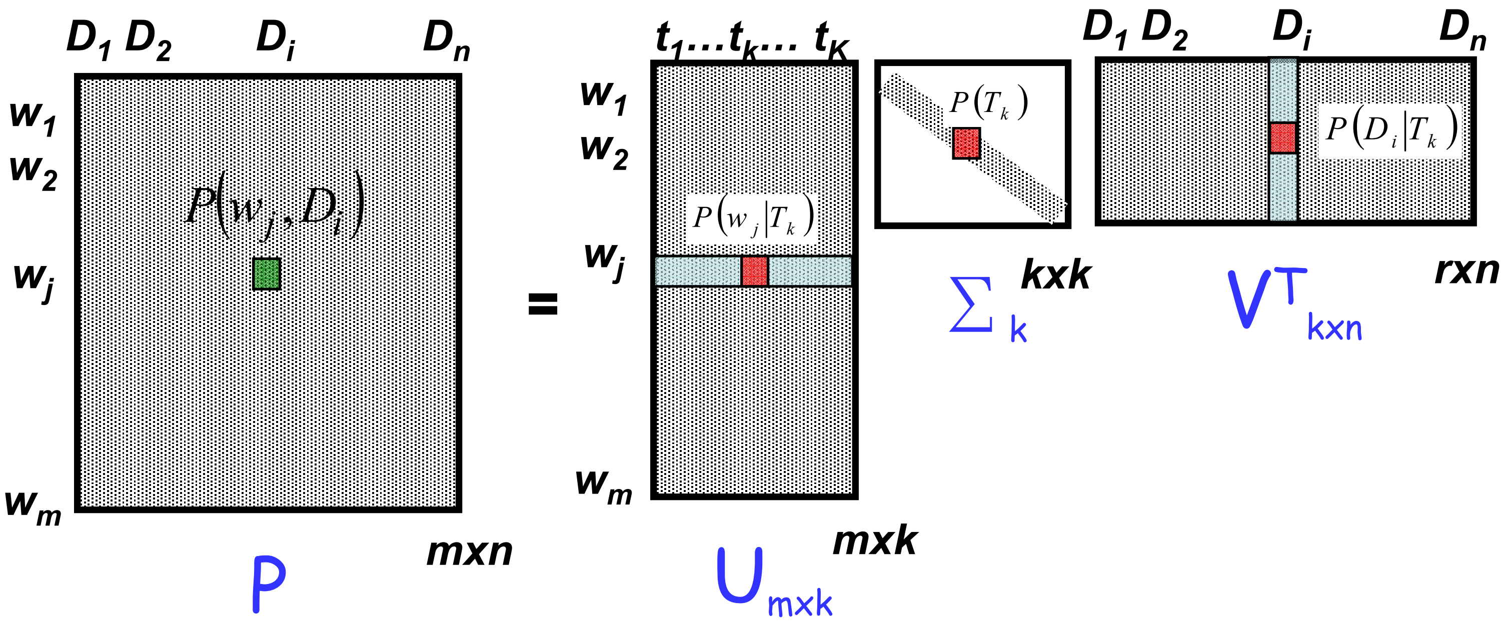
$$P(w_j, D_i) = \sum_{T_k} P(w_j, T_k, D_i) = \sum_{T_k} P(w_j | T_k, D_i) P(T_k, D_i)$$

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

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

# Probabilistic Latent Semantic Analysis (cont.)

- Probabilistic Latent Semantic 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
  - Two ways to smoothen empirical distribution with PLSI
    - Combine the cosine score with that of the vector space model (so does LSI)

      **PLSI-U*** (See next slide)
    - Combine the multinomials individually

      **PLSI-U***

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

$$P_{PLSI-Q^*}(\omega_j \mid D_i) = \lambda P_{Empirical}(\omega_j \mid D_i) + (1-\lambda) P_{PLSI}(\omega_j \mid D_i)$$

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

$$P_{PLSI-Q^*}(Q \mid D_i) = \prod_{\omega_j \in Q} \left( \lambda P_{Empirical}(\omega_j \mid D_i) + (1-\lambda) P_{PLSI}(\omega_j \mid D_i) \right)$$

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 $P(T_k \mid Q)$ 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 re-weight the different model components (dimensions) by

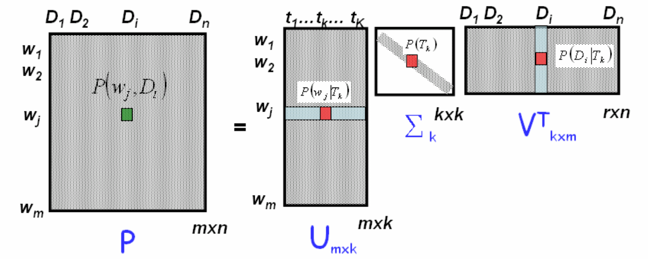
$$R_{PLSI-Q^*}(Q, D_i) = \frac{\sum_k P(T_k|Q)P(T_k|D_i)}{\sqrt{\sum_k P(T_k|Q)^2}\sqrt{\sum_k P(T_k|D_i)^2}} \quad \text{, where } P(T_k|Q) = \frac{\sum_{q_n \in Q} n(q_n, Q) P(T_k|q_n, Q)}{|Q|}$$

online folded-in

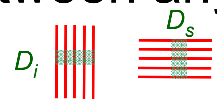
$$\tilde{R}_{PLSI-Q^*}(Q, D_i) = \lambda \cdot R_{PLSI}(Q, D_i) + (1-\lambda) \cdot R_{VSM}(\vec{Q}, \vec{D}_i)$$

# Probabilistic Latent Semantic Analysis (cont.)

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




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




$A'^TA'=(U'\Sigma'V'^T)^{T'}(U'\Sigma'V'^T)=V'\Sigma'^{T}U^TU'\Sigma'V'^T=(V'\Sigma')(V'\Sigma')^T$

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

  - PLAS mimics LSI in similarity measure

$\hat{D}_i$ and $\hat{D}_s$ are row vectors

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

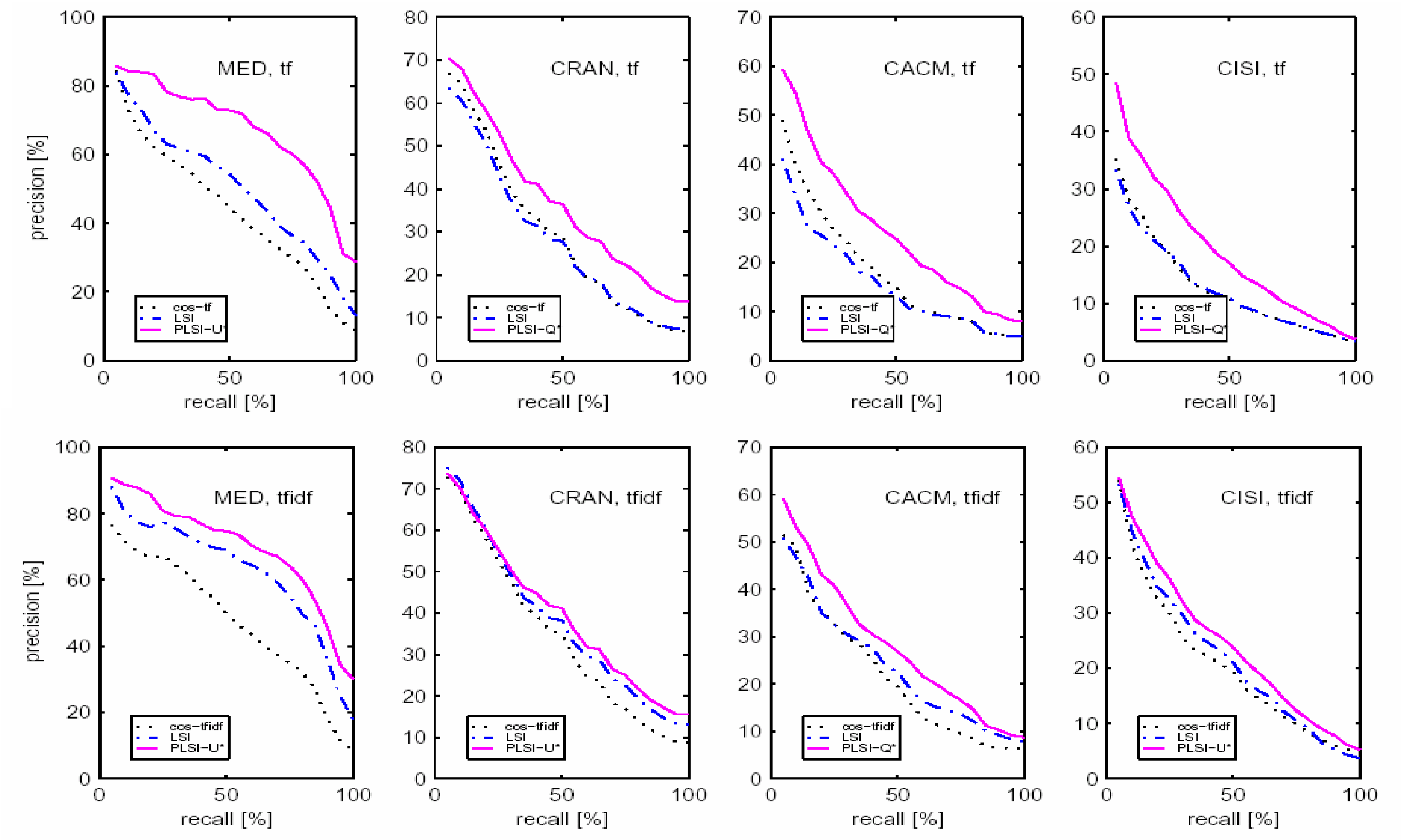
$$=\frac{\sum_k P(T_k|D_i)\cancel{P(D_i)}P(T_k|D_s)\cancel{P(D_s)}}{\sqrt{\sum_k [P(T_k|D_i)\cancel{P(D_i)}]^2}\sqrt{\sum_k [P(T_k|D_s)\cancel{P(D_s)}]^2}}$$

$$=\frac{\sum_k P(T_k|D_i)P(T_k|D_s)}{\sqrt{\sum_k P(T_k|D_i)^2}\sqrt{\sum_k P(T_k|D_s)^2}}$$

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

# Probabilistic Latent Semantic Analysis (cont.)

- Experimental Results



# Comparisons

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

256 Topics　　　　256 Topics

| Retrieval Model | **TMM** | **HMM** | PLSA | VSM | LSI |
|---|---|---|---|---|---|
| TD | **0.7870** | **0.7174** | 0.6513 | 0.6505 | 0.6440 |
| SD | **0.7852** | **0.7156** | 0.5989 | 0.6216 | 0.6390 |

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

# LSA: SVDLIBC

- Doug Rohde's SVD C Library version 1.3 is based on the SVDPACKC library
- Download it at http://tedlab.mit.edu/~dr/

# HW: Latent Semantic Analysis (LSA)

- Given a sparse term-document matrix
  - E.g., 4 terms and 3 docs

$$\begin{array}{cc} & \text{Doc} \\ \text{Term} & \begin{Bmatrix} 2.3 & 0.0 & 4.2 \\ 0.0 & 1.3 & 2.2 \\ 3.8 & 0.0 & 0.5 \\ 0.0 & 0.0 & 0.0 \end{Bmatrix} \end{array}$$

| Row #Tem | Col. # Doc | Nonzero entries |
|---|---|---|
| 4 | 3 | 6 |

| | |
|---|---|
| 2 | 2 nonzero entries at **Col 0** |
| 0 2.3 | Col 0, Row 0 |
| 2 3.8 | Col 0, Row 2 |
| 1 | 1 nonzero entry at **Col 1** |
| 1 1.3 | Col 1, Row 1 |
| 3 | 3 nonzero entry at **Col 2** |
| 0 4.2 | Col 2, Row 0 |
| 1 2.2 | Col 2, Row 1 |
| 2 0.5 | Col 2, Row 2 |

  - Each entry can be weighted by *TF*x*IDF* score

- Perform SVD to obtain term and document vectors represented in the latent semantic space

- Evaluate the information retrieval capability of the LSA approach by using varying sizes (e.g., 100, 200, ..,600 etc.) of LSA dimensionality

# HW: Latent Semantic Analysis (cont.)

- Example: term-document matrix

Indexing Term no. | Doc no. | Nonzero entries

```
51253 2265 218852
77
508 7.725771
596 16.213399
612 13.080868
709 7.725771
713 7.725771
744 7.725771
1190 7.725771
1200 16.213399
1259 7.725771
……
```

- SVD command (IR_svd.bat)

**svd** -r st -o LSA100 -d 100 Term-Doc-Matrix

sparse matrix input | prefix of output files | No. of reserved eigenvectors | name of sparse matrix input

output → **LSA100-Ut**, **LSA100-S**, **LSA100-Vt**

# HW: Latent Semantic Analysis (cont.)

- **LSA100-Ut**

51253 words

100 51253

$\underbrace{\boxed{0.003}\ 0.001\ \ldots\ldots\ldots}$

$\boxed{0.002}\ 0.002\ \ldots\ldots\ldots$

word vector ($u^T$): 1x100

- **LSA100-Vt**

2265 docs

100 2265

$\underbrace{\boxed{0.021}\ 0.035\ \ldots\ldots\ldots}$

$\boxed{0.012}\ 0.022\ \ldots\ldots\ldots$

doc vector ($v^T$): 1x100

- **LSA100-S**

100

2686.18

829.941

559.59

….

100 eigenvalues

# HW: Latent Semantic Analysis (cont.)

- Fold-in a new *m*x1 query vector

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

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

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