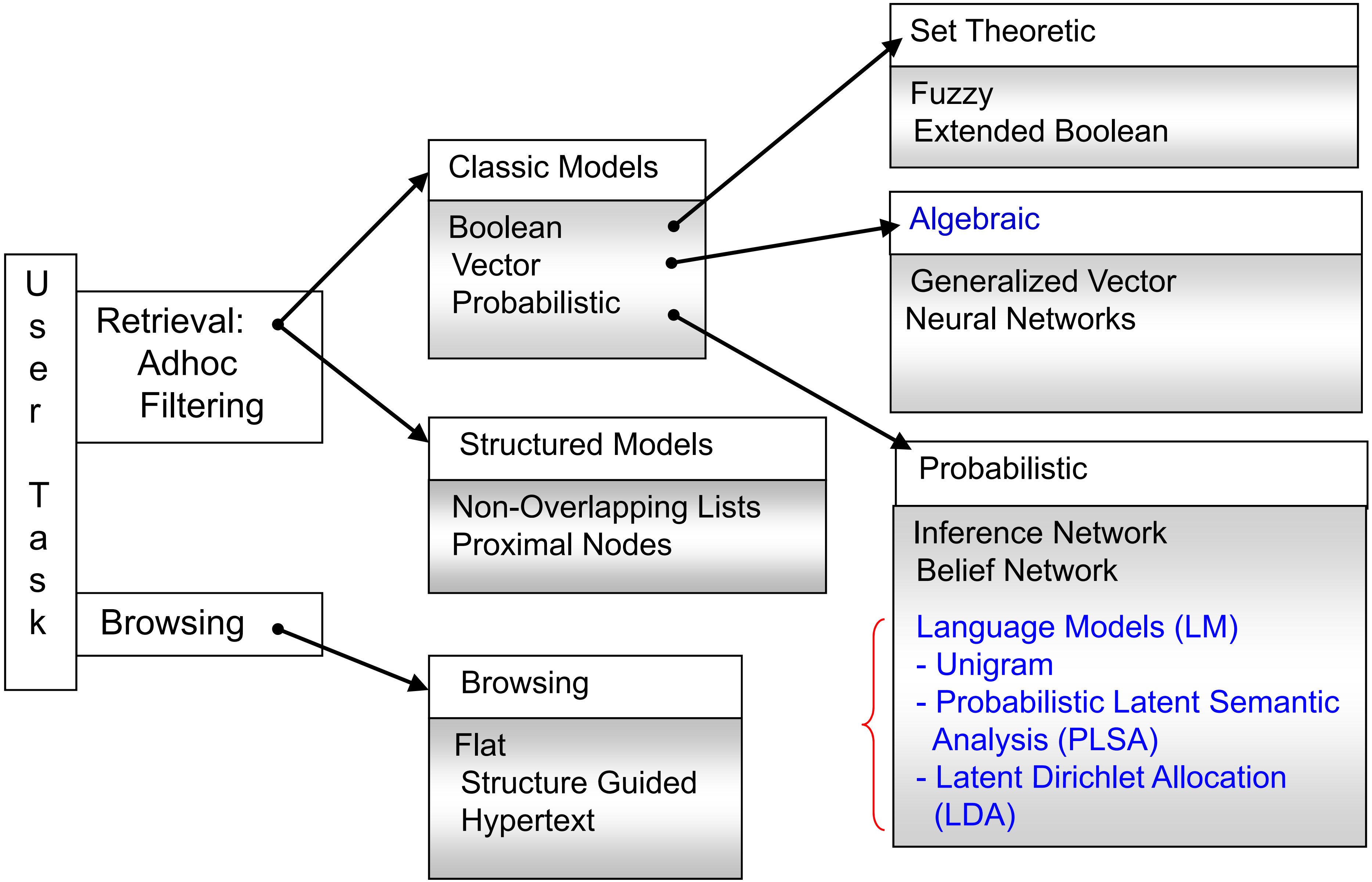
Language Models for Information Retrieval

Berlin Chen
Department of Computer Science & Information Engineering
National Taiwan Normal University

References:

1. W. B. Croft and J. Lafferty (Editors). Language Modeling for Information Retrieval. July 2003
2. X. Liu and W.B. Croft, Statistical Language Modeling For Information Retrieval, the Annual Review of Information Science and Technology, vol. 39, 2005
3. Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, Introduction to Information Retrieval, Cambridge University Press, 2008. (Chapter 12)
4. D. A. Grossman, O. Frieder, Information Retrieval: Algorithms and Heuristics, Springer, 2004 (Chapter 2)
5. C.X. Zhai. Statistical Language Models for Information Retrieval (Synthesis Lectures Series on Human Language Technologies). Morgan & Claypool Publishers, 2008

Taxonomy of Classic IR Models



Statistical Language Models (1/2)

- A probabilistic mechanism for "generating" a piece of text
  - Defines a distribution over all possible word sequences

$$W = w_1 w_2 \ldots w_L$$

$$P(W) = ?$$

- What is LM Used for ?
  - Speech recognition
  - Spelling correction
  - Handwriting recognition
  - Optical character recognition
  - Machine translation
  - Document classification and routing
  - Information retrieval …

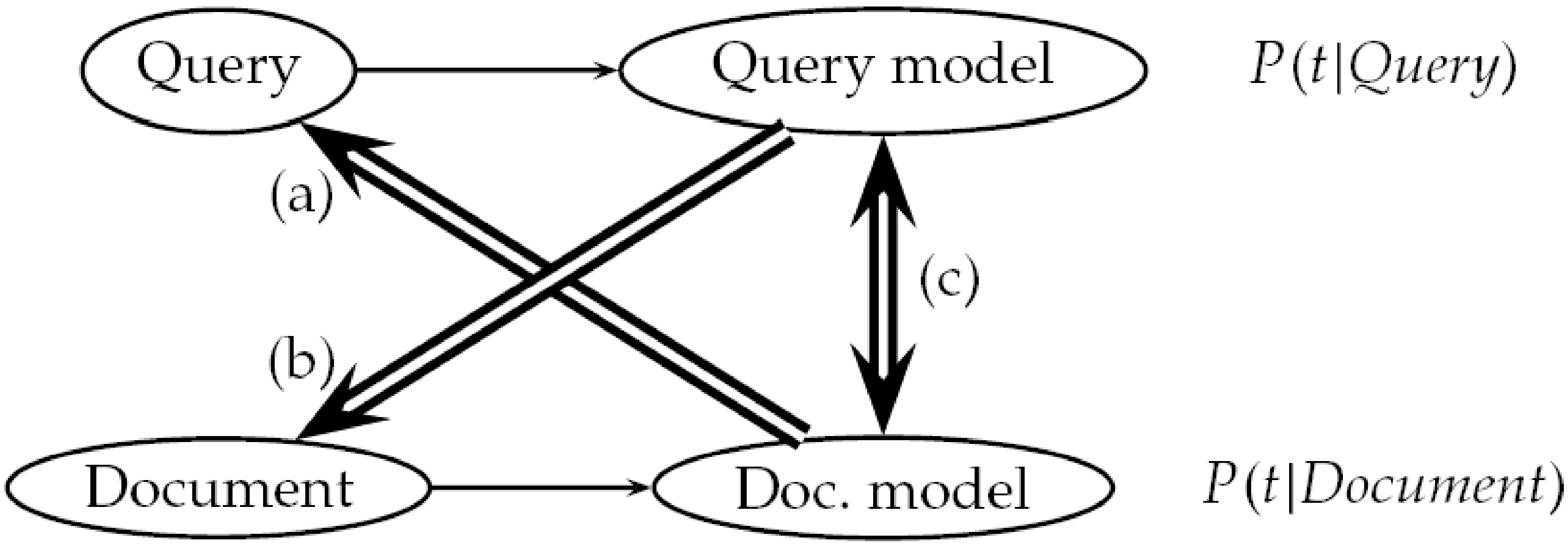
Statistical Language Models (2/2)

- (Statistical) language models (LM) have been widely used for speech recognition and language (machine) translation for more than twenty years

- However, their use for information retrieval started only in 1998 [Ponte and Croft, SIGIR 1998]
  - Basically, a query is considered generated from an "ideal" document that satisfies the information need
  - The system's job is then to estimate the likelihood of each document in the collection being the ideal document and rank then accordingly (in decreasing order)

Ponte and Croft. A language modeling approach to information retrieval. *SIGIR 1998*

Three Ways of Developing LM Approaches for IR



(a) Query likelihood

(b) Document likelihood

(c) Model comparison

literal term matching
or concept matching

Query-Likelihood Language Models

- Criterion: Documents are ranked based on Bayes (decision) rule

$$P(D|Q) = \frac{P(Q|D)P(D)}{P(Q)}$$

  - $P(Q)$ is the same for all documents, and can be ignored
  - $P(D)$ might have to do with authority, length, genre, etc.
    - There is no general way to estimate it
    - Can be treated as uniform across all documents

- Documents can therefore be ranked based on

document model

$$P(Q|D) \quad \left(\text{or denoted as } P(Q|\mathrm{M}_D)\right)$$

  - The user has a prototype (ideal) document in mind, and generates a query based on words that appear in this document
  - A document D is treated as a model M_D to predict (generate) the query

Another Criterion: Maximum Mutual Information

- Documents can be ranked based their mutual information with the query (in decreasing order)

$$
\begin{aligned}
MI(Q,D) &= \log \frac{P(Q,D)}{P(Q)P(D)} \\
&= \log P(Q|D) - \underbrace{\log P(Q)}_{\text{being the same for all documents, and hence can be ignored}}
\end{aligned}
$$

- Document ranking by mutual information (MI) is equivalent that by likelihood

$$
\arg\max_{D} MI(Q,D) \overset{\text{rank}}{=} \arg\max_{D} P(Q|D)
$$

Yet Another Criterion: Minimum KL Divergence

- Documents are ranked by Kullback-Leibler (KL) divergence (in increasing order)

Query model　　Document model

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

$$= \boxed{\sum_w P(w|Q)\log P(w|Q)} - \boxed{\sum_w P(w|Q)\log P(w|D)}$$

The same for all document
=> can be disregarded

Cross entropy between the language models of a query and a document

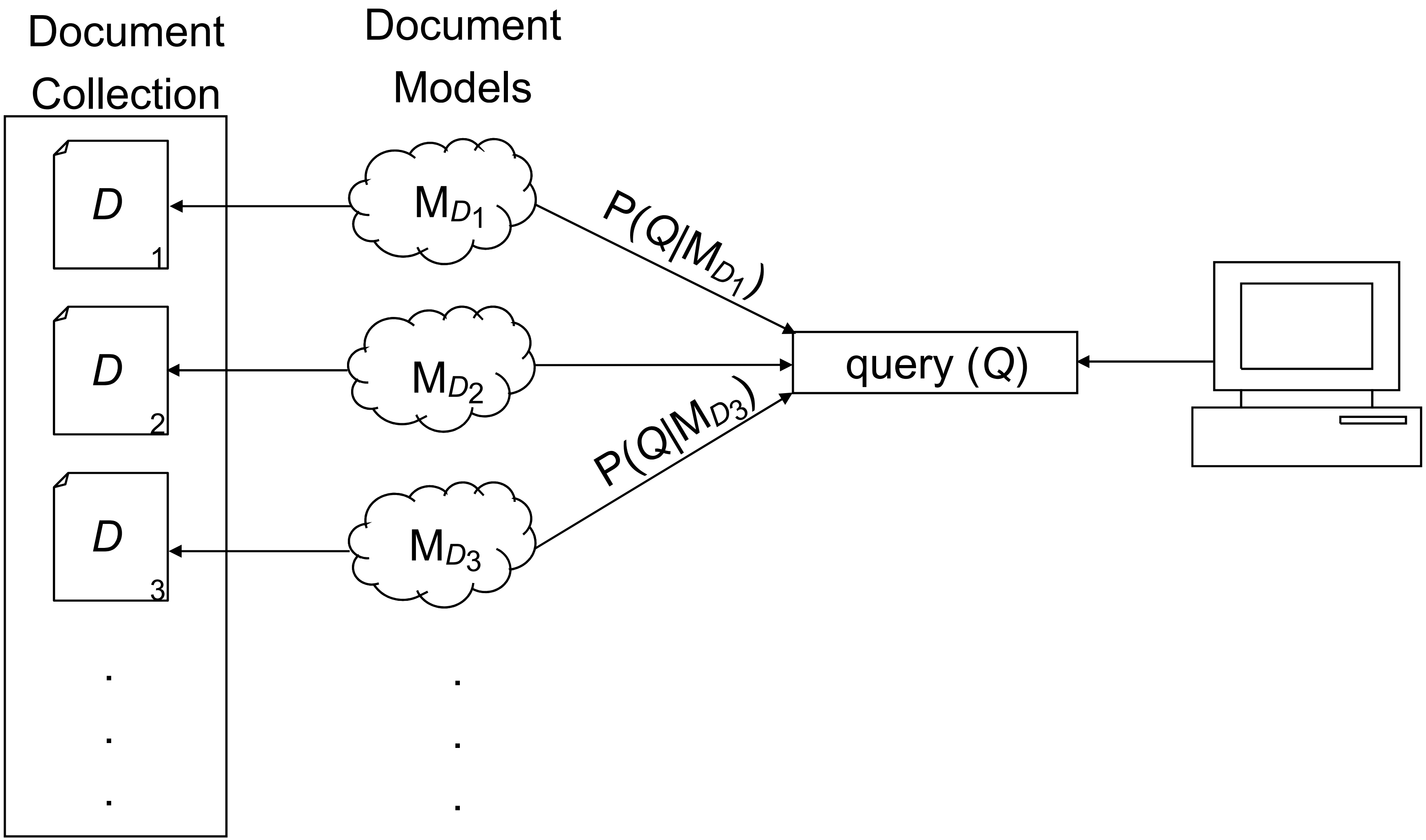
Equivalent to ranking in decreasing order of

$$\sum_w P(w|Q)\log P(w|D)$$

Relevant documents deemed to have lower cross entropies

$$\overset{\text{rank}}{=} \sum_w c(w,Q)\log P(w|D) = P(Q|D)$$

Schematic Depiction



n-grams

- Multiplication (Chain) rule

$$P(w_1 w_2 .... w_L) = P(w_1) P(w_2 | w_1) P(w_3 | w_1 w_2) \cdots P(w_L | w_1 w_2 \ldots w_{L-1})$$

  – Decompose the probability of a sequence of events into the probability of each successive events conditioned on earlier events

- *n*-gram assumption
  – Unigram

$$P(w_1 w_2 .... w_L) = P(w_1) P(w_2) P(w_3) \cdots P(w_L)$$

    - Each word occurs independently of the other words
    - The so-called "bag-of-words" model (e.g., how to distinguish "street market" from "market street)

  – Bigram

$$P(w_1 w_2 .... w_L) = P(w_1) P(w_2 | w_1) P(w_3 | w_2) \cdots P(w_L | w_{L-1})$$

  – Most language-modeling work in IR has used unigram models
    - IR does not directly depend on the structure of sentences

Unigram Model (1/4)

- The likelihood of a query $Q = w_1 w_2 .... w_L$ given a document D

$$P(Q|\mathrm{M}_D) = P(w_1|\mathrm{M}_D)P(w_2|\mathrm{M}_D)\cdots P(w_L|\mathrm{M}_D)$$
$$= \prod_{i=1}^{L} P(w_i|\mathrm{M}_D)$$

  - Words are conditionally independent of each other given the document
  - How to estimate the probability of a (query) word given the document $P(w|\mathrm{M}_D)$?

- Assume that words follow a multinomial distribution given the document

permutation is considered here

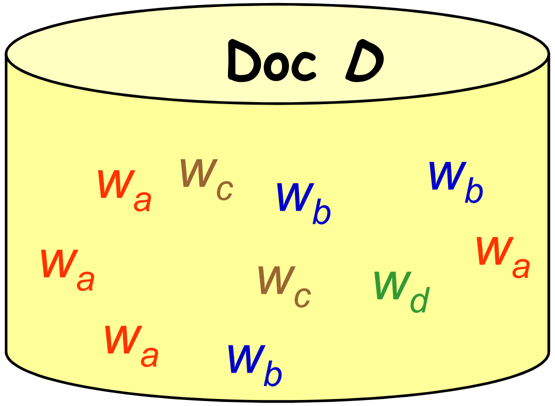
$$P(c(w_1),\ldots,c(w_V)|\mathrm{M}_D) = \frac{\left(\sum_{j=1}^{V} c(w_j)\right)!}{\prod_{i=1}^{V}(c(w_i)!)} \prod_{i=1}^{V} \lambda_{w_i}^{c(w_i)}$$

where $c(w_i)$: the number of times a word occurs

$$\lambda_{w_i} = P(w_i|\mathrm{M}_D), \quad \sum_{i=1}^{V} \lambda_{w_i} = 1$$

Unigram Model (2/4)

- Use each document itself a sample for estimating its corresponding unigram (multinomial) model
  - If Maximum Likelihood Estimation (MLE) is adopted



$$\hat{P}(w_i|\mathrm{M}_D)=\frac{c(w_i,D)}{|D|}$$

where

$c(w_i,D)$: number of times w_i occurs in D

$|D|$: length of D, $\sum_i c(w_i,D)=|D|$

$P(w_b|M_D)=0.3$

$P(w_c|M_D)=0.2$

$P(w_d|M_D)=0.1$

$P(w_e|M_D)=0.0$

$P(w_f|M_D)=0.0$

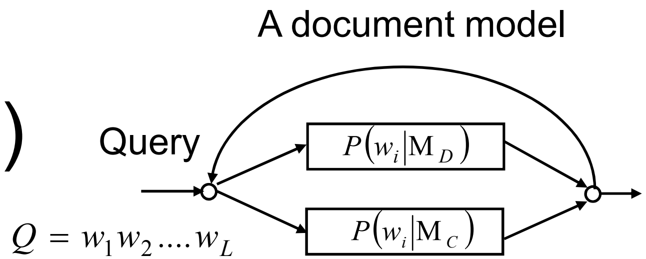
The zero-probability problem

If w_e and w_f do not occur in D

then $P(w_e|M_D)= P(w_f|M_D)=0$

This will cause a problem in predicting the query likelihood (See the equation for the query likelihood in the preceding slide)

Unigram Model (3/4)



- Smooth the document-specific unigram model with a collection model (two states, or a mixture of two multinomials)

$$P(Q|\mathrm{M}_D) = \prod_{i=1}^{L} \left[\lambda \cdot P(w_i|\mathrm{M}_D) + (1-\lambda) \cdot P(w_i|\mathrm{M}_C) \right]$$

- The role of the collection unigram model $P(w_i|\mathrm{M}_C)$
  - Help to solve zero-probability problem
  - Help to differentiate the contributions of different missing terms in a document (global information like IDF ?)

$$P(w_i|\mathrm{M}_C) = \frac{c(w_i, Collection)}{\sum_{w_l} c(w_l, Collection)} \quad \text{or} \quad \boxed{\frac{\frac{N}{n_i}}{\sum_{w_l} \frac{N}{n_l}}}$$

N : number of doc in the collection

n_i : number of doc in the collection containing w_i

Normalized doc freq

- The collection unigram model can be estimated in a similar way as what we do for the document-specific unigram model

Unigram Model (4/4)

- An evaluation on the Topic Detection and Tracking (TDT) corpora
  - Language Model

mAP		Unigram	Unigram+Bigram
TDT2	TQ/TD	**0.6327**	0.5427
	TQ/SD	0.5658	0.4803
TDT3	TQ/TD	**0.6569**	0.6141
	TQ/SD	0.6308	0.5808

  - Vector Space Model

*m*AP		Unigram	Unigram+Bigram
TDT2	TQ/TD	0.5548	**0.5623**
	TQ/SD	0.5122	**0.5225**
TDT3	TQ/TD	0.6505	**0.6531**
	TQ/SD	0.6216	0.6233

$$P_{Unigram}(Q|M_D) = \prod_{i=1}^{L}\left[\lambda \cdot P(w_i|M_D) + (1-\lambda)\cdot P(w_i|M_C)\right]$$

$$\begin{aligned} P_{Unigram+Bigram}(Q|M_D) &= \prod_{i=1}^{L}\left[\lambda_1 \cdot P(w_i|M_D) + \lambda_2 \cdot P(w_i|M_C)\right. \\ &\quad \lambda_3 \cdot P(w_i|w_{i-1}, M_D) + \\ &\quad \left.(1-\lambda_1-\lambda_2-\lambda_3)\cdot P(w_i|w_{i-1}, M_C)\right] \end{aligned}$$

- Consideration of contextual information (Higher-order language models, e.g., bigrams) will not always lead to improved performance

Statistical Translation Model (1/2)

Berger & Lafferty (1999)

- A query Q is viewed as a translation or distillation from a document D
  - That is, the similarity measure is computed by estimating the probability that the query would have been generated as a translation of that document

$$sim(Q,D) = P(Q|D) = \prod_{q \in Q} P_{\text{Trans}}(q|D)^{c(q,Q)} = \prod_{q \in Q} \sum_{w \in D} \left[\underline{P(q|w)} P(w|D) \right]^{c(q,Q)}$$

word-to-word translation

- Assumption of context-independence (the ability to handle the ambiguity of word senses is limited)
- However, it the capability of handling the issues of synonymy (multiple terms having similar meaning) and polysemy (the same term having multiple meanings)

$$\hat{P}(Q|D) = \prod_{q \in Q} \left[\lambda P_{\text{Trans}}(q|D) + (1-\lambda)(q|\mathrm{M}_C) \right]^{c(q,Q)}$$

A. Berger and J. Lafferty. Information retrieval as statistical translation. *SIGIR 1999*

Statistical Translation Model (2/2)

- Weakness of the statistical translation model
  - The need of a large collection of training data for estimating translation probabilities, and inefficiency for ranking documents
- Jin et al. (2002) proposed a "Title Language Model" approach to capture the intrinsic document to query translation patterns
  - Queries are more like titles than documents (queries and titles both tend to be very short and concise descriptions of information, and created through a similar generation process)
  - Train the statistical translation model based on the document-title pairs in the whole collection

$$M^* = \arg\max_M \prod_{j=1}^{N} P_M\left(T_j \middle| D_j\right) = \arg\max_M \prod_{j=1}^{N} \prod_{t \in T_j} P_M\left(t \middle| D_j\right)$$

$$\approx \arg\max_M \prod_{j=1}^{N} \prod_{t \in T_j} \left(\sum_{w \in D} \left[P_M(t|w) P(w|D) \right]^{c(t, T_j)} \right)$$

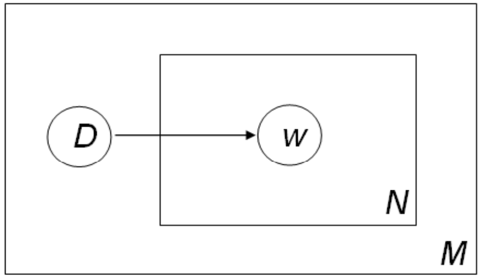
R. Jin et al. Title language model for information retrieval. *SIGIR 2002*

Probabilistic Latent Semantic Analysis (PLSA)

Hofmann (1999)

- Also called The Aspect Model, Probabilistic Latent Semantic Indexing (PLSI)
  - Graphical Model Representation (a kind of Bayesian Networks)

Language (unigram) model



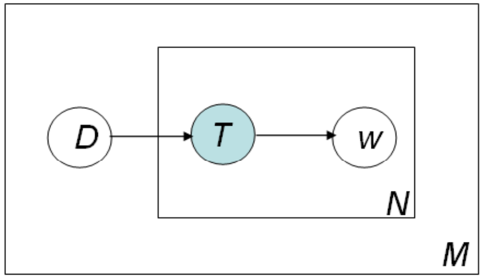
$$sim(Q,D) = P(D|Q) = \frac{P(Q|D)P(D)}{P(Q)}$$

$$\propto P(Q|D)P(D)$$

$$\approx P(Q|D)$$

$$= \prod_{w \in Q} \left[\lambda \cdot P(w|M_D) + (1-\lambda) \cdot P(w|M_C)\right]^{c(w,Q)}$$

PLSA



N: number of distinct in the vocabulary

M: number of documents in the collection

: observed variable

: latent variable

$$sim(Q,D) = P(Q|D) = \prod_{w \in Q} P(w|D)^{c(w,Q)}$$

$$= \prod_{w \in Q} \left[\sum_{k=1}^{K} P(w,T_k|D)\right]^{c(w,Q)}$$

$$= \prod_{w \in Q} \left[\sum_{k=1}^{K} P(w|T_k)P(T_k|D)\right]^{c(w,Q)}$$

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

T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning* 2001

PLSA: Formulation

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

PLSA: Assumptions

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

$$sim(Q, D) = P(Q|D) = \prod_{w} P(w|D)^{c(w,Q)}$$

- **Conditional independent**: the doc D and word w are independent conditioned on the state of the associated latent variable T_k

$$P(w, D|T_k) \approx P(w|T_k)P(D|T_k)$$

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

PLSA: Training (1/2)

- Probabilities are estimated by maximizing the collection likelihood using the Expectation-Maximization (EM) algorithm

$$L_C = \sum_D \sum_w c(w, D) \log P(w|D)$$

$$= \sum_D \sum_w c(w, D) \log \left[\sum_{T_k} P(w|T_k) P(T_k|D) \right]$$

EM tutorial:

- 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

PLSA: Training (2/2)

- E (expectation) step

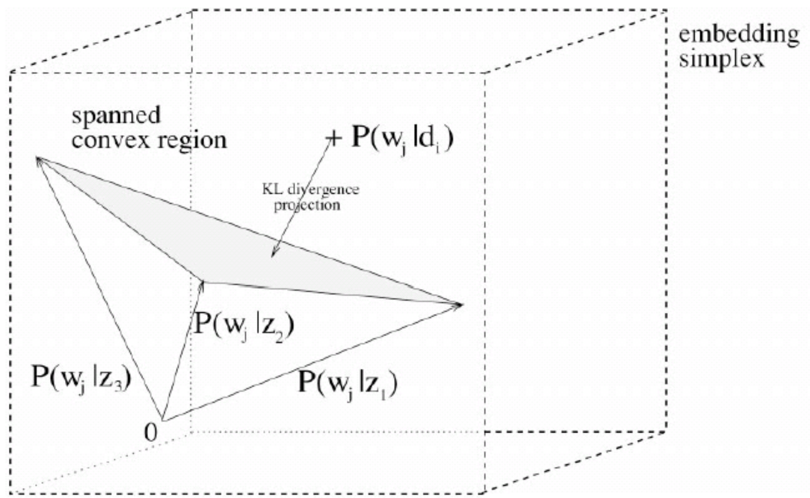
$$P(T_k|w,D) = \frac{P(w|T_k)P(T_k|D)}{\sum_{T_k} P(w|T_k)P(T_k|D)}$$

- M (Maximization) step

$$\hat{P}(w|T_k) = \frac{\sum_D c(w,D)P(T_k|w,D)}{\sum_w \sum_D c(w,D)P(T_k|w,D)}$$

$$\hat{P}(T_k|D) = \frac{\sum_w c(w,D)P(T_k|w,D)}{\sum_{w'} c(w',D)}$$

PLSA: Latent Probability Space (1/2)



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

Dimensionality K=128 (latent classes)

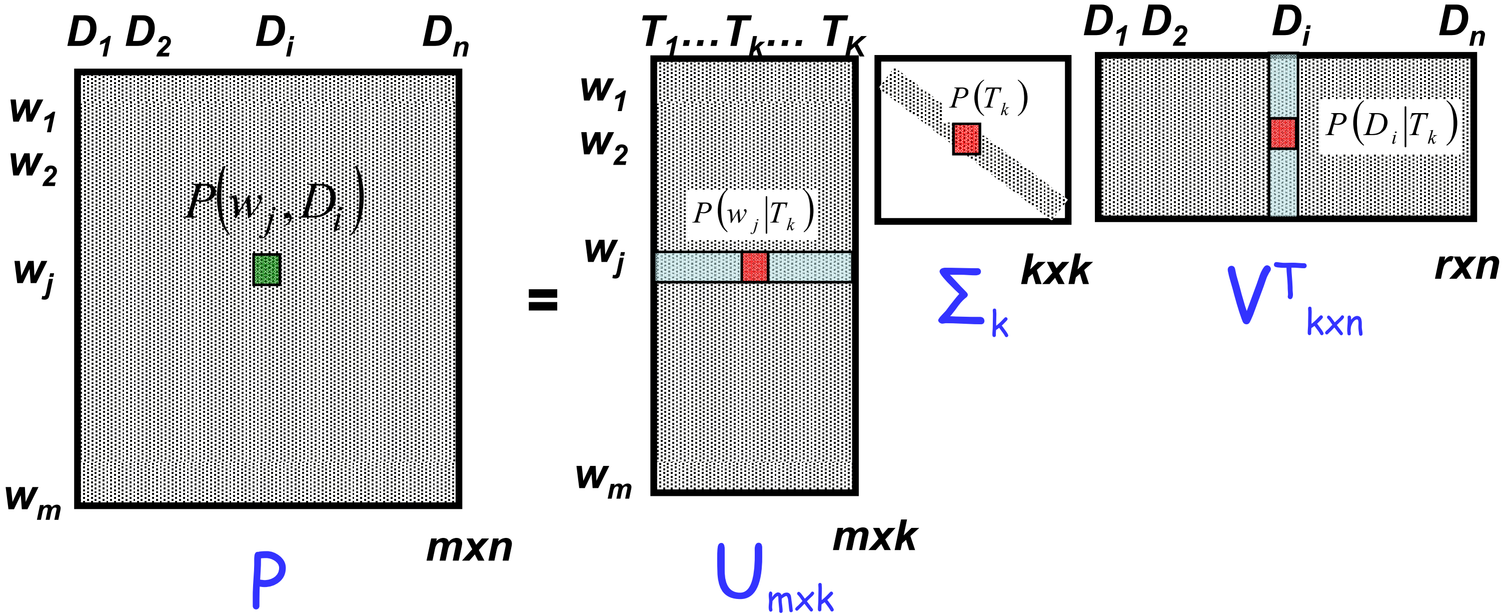
Aspect 1	Aspect 2	Aspect 3	Aspect 4
imag	video	region	speaker
SEGMENT	sequenc	contour	speech
textur	motion	boundari	recogni
color	frame	descrip	signal
tissu	scene	imag	train
brain	SEGMENT	SEGMENT	hmm
slice	shot	precis	sourc
cluster	imag	estim	speakerindepend
mri	cluster	pixel	SEGMENT
algorithm	visual	paramet	sound
medical imaging	image sequence analysis	context of contour boundary detection	phonetic segmentation

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

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

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

PLSA: Latent Probability Space (2/2)



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

PLSA: One more example on TDT1 dataset

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

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

PLSA: Experiment Results (1/4)

- Experimental Results
  - Two ways to smoothen empirical distribution with PLSA
    - Combine the cosine score with that of the vector space model (so does LSA)

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

      PLSA-Q*

$$P_{PLSA}(w|D)=\sum_{k=1}^{K}P(w|T_k)P(T_k|D)$$

$$P_{PLSA-Q*}(w|D)=\lambda\cdot \boxed{P_{Empirical}(w|D)}+(1-\lambda)\cdot P_{PLSA}(w|D)$$

$$P_{Empirical}(w|D)=\frac{c(w,D)}{c(D)}$$

$$P_{PLSA-Q*}(Q|D)=\prod_{w\in Q}\left(\lambda\cdot P_{Empirical}(w|D)+(1-\lambda)\cdot P_{PLSA}(w|D)\right)^{c(w,D)}$$

Both provide almost identical performance

- It's not known if PLSA ($P_{PLSA}(w|D)$) was used alone

PLSA: Experiment Results (2/4)

PLSA-U*

- Use the low-dimensional representation $P(T_k \mid Q)$ and $P(T_k \mid D)$ (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

$$R_{PLSA-U*}(Q,D) = \frac{\sum_k P(T_k|Q)P(T_k|D)}{\sqrt{\sum_k P(T_k|Q)^2}\sqrt{\sum_k P(T_k|D)^2}}$$

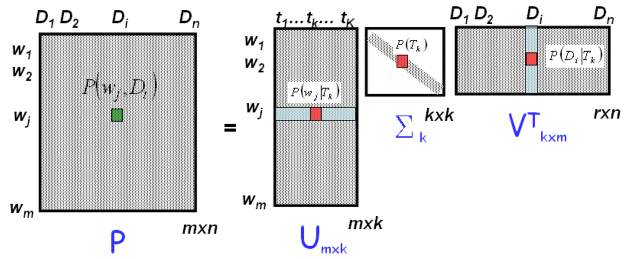
$$\text{, where } P(T_k|Q) = \frac{\sum_{w \in Q} c(w,Q)\, P(T_k|w,Q)}{\sum_{w' \in Q} c(w',Q)}$$

online folded-in

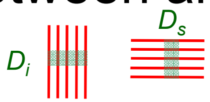
$$\tilde{R}_{PLSA-U*}(Q,D) = \lambda \cdot R_{PLSA-U*}(Q,D) + (1-\lambda) \cdot R_{VSM}(\vec{Q},\vec{D})$$

PLSA: Experiment Results (3/4)

- 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 LSA, the relations between any two docs can be formulated as



$$A'^T A' = (U'\Sigma'V'^T)^T (U'\Sigma'V'^T) = V'\Sigma'^T U^T U'\Sigma'V'^T = (V'\Sigma')(V'\Sigma')^T$$

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

  - PLSA mimics LSA in similarity measure

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

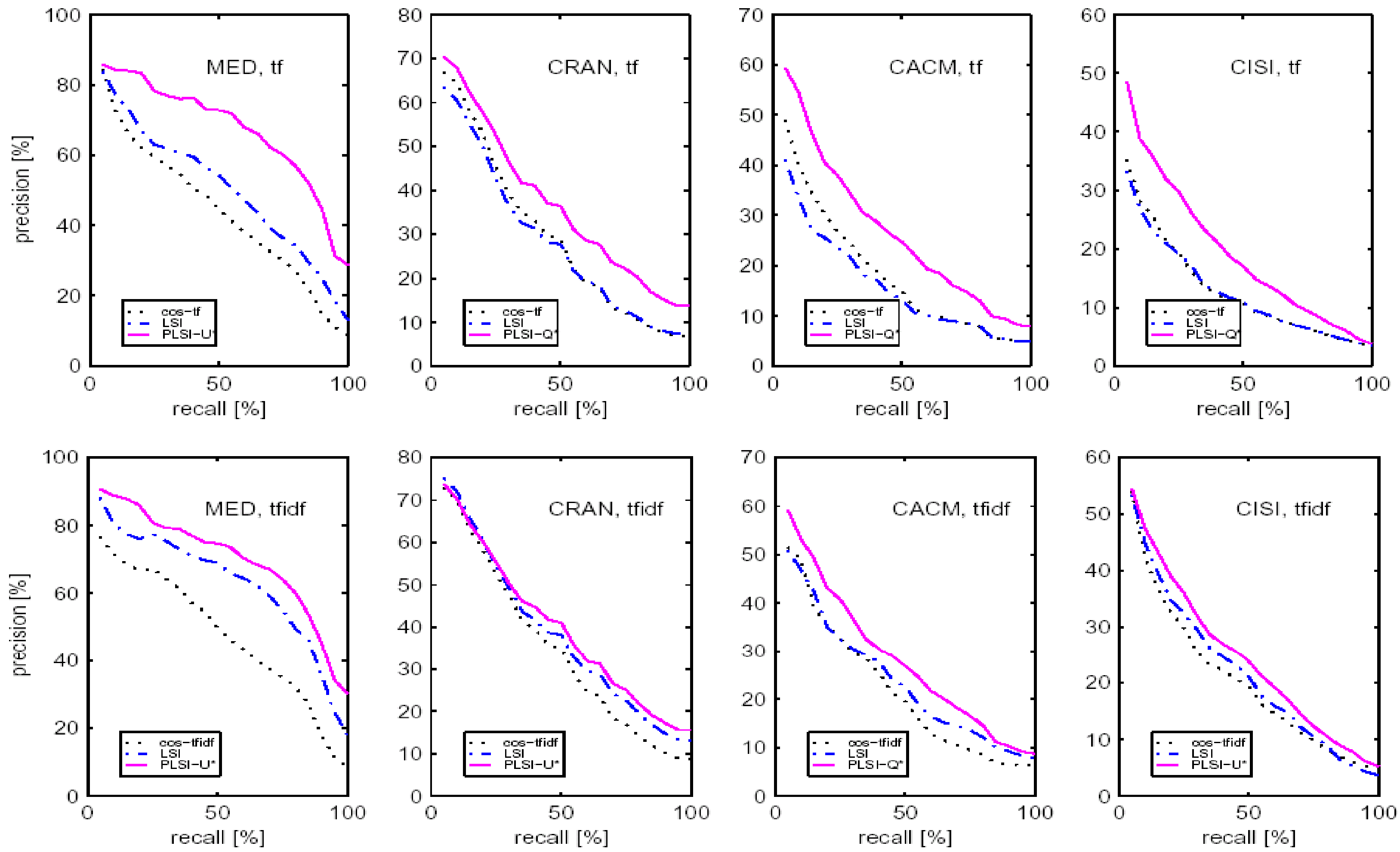
$$R_{PLSI-Q^*}(D_i,D_s)=\frac{\sum_k P(D_i|T_k)P(T_k)P(T_k)P(D_s|T_k)}{\sqrt{\sum_k [P(D_i|T_k)P(T_k)]^2}\sqrt{\sum_k [P(D_i|T_k)P(T_k)]^2}}$$

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

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

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

PLSA: Experiment Results (4/4)



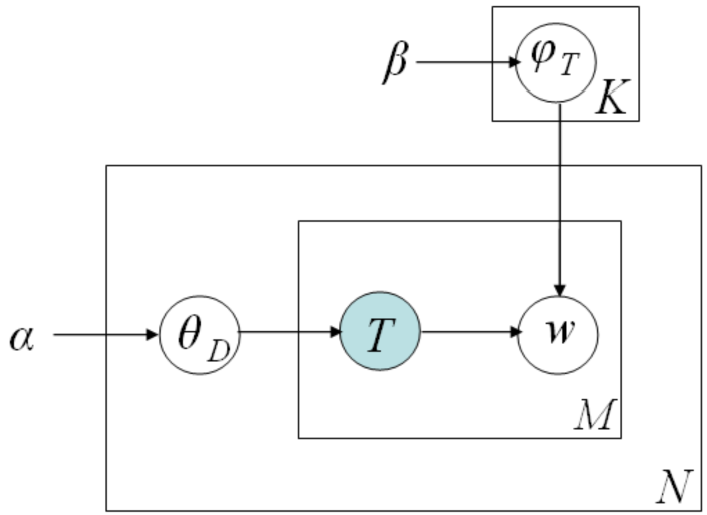
PLSA vs. LSA

- Decomposition/Approximation
  - **LSA**: 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
  - LSA: SVD decomposition
  - PLSA: EM training, is time-consuming for iterations ?
  - The model complexity of Both LSA and PLSA grows linearly with the number of training documents
    - There is no general way to estimate or predict the vector representation (of LSA) or the model parameters (of PLSA) for a newly observed document

Latent Dirichlet Allocation (LDA) (1/2)

Blei et al. (2003)

- The basic generative process of LDA closely resembles PLSA; however,
  - In PLSA, the topic mixture $P(T_k|D)$ is conditioned on each document ($P(T_k|D)$ is fixed, unknown)
  - While in LDA, the topic mixture $P(T_k|D)$ is drawn from a Dirichlet distribution, so-called the conjugate prior, ($P(T_k|D)$ is unknown and follows a probability distribution)



Process of generating a corpus with LDA

1) Pick a multinomial distribution φ_T for each topic T from a Dirichlet distribution with parameter β

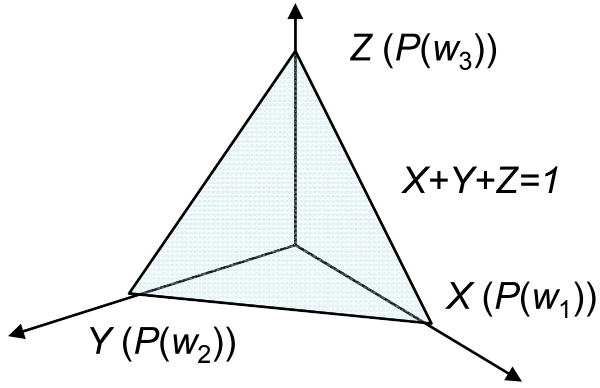
2) Pick a multinomial distribution θ_D for each docu D from a Dirichlet distribution with parameter α

3) Pick a topic $T \in \{1,2,\cdots,K\}$ from a multinomial distribution with parameter θ_D

4) Pick a *w*ord from a multinomial distribution with parameter φ_T

Blei et al. Latent Dirichlet allocation. Journal of Machine Learning Research, 2003

Latent Dirichlet Allocation (2/2)



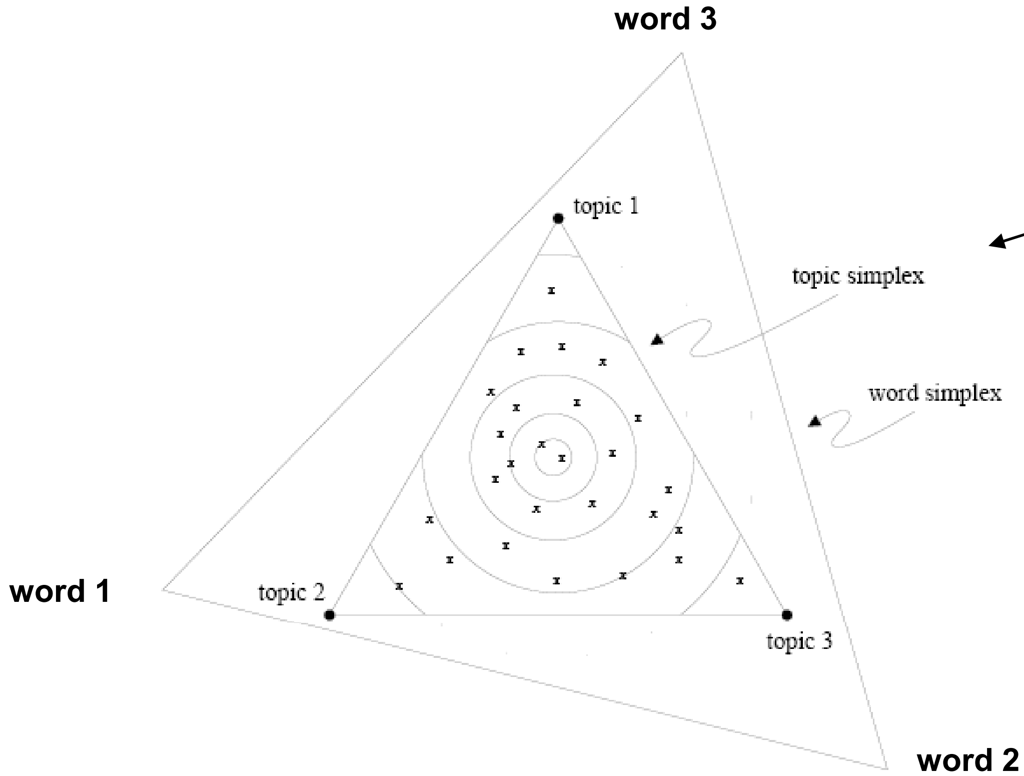


Figure 4: The topic simplex for three topics embedded in the word simplex for three words. The corners of the word simplex correspond to the three distributions where each word (respectively) has probability one. The three points of the topic simplex correspond to three different distributions over words. The mixture of unigrams places each document at one of the corners of the topic simplex. The pLSI model induces an empirical distribution on the topic simplex denoted by x. LDA places a smooth distribution on the topic simplex denoted by the contour lines.

Word Topical Mixture Models (WTMM)

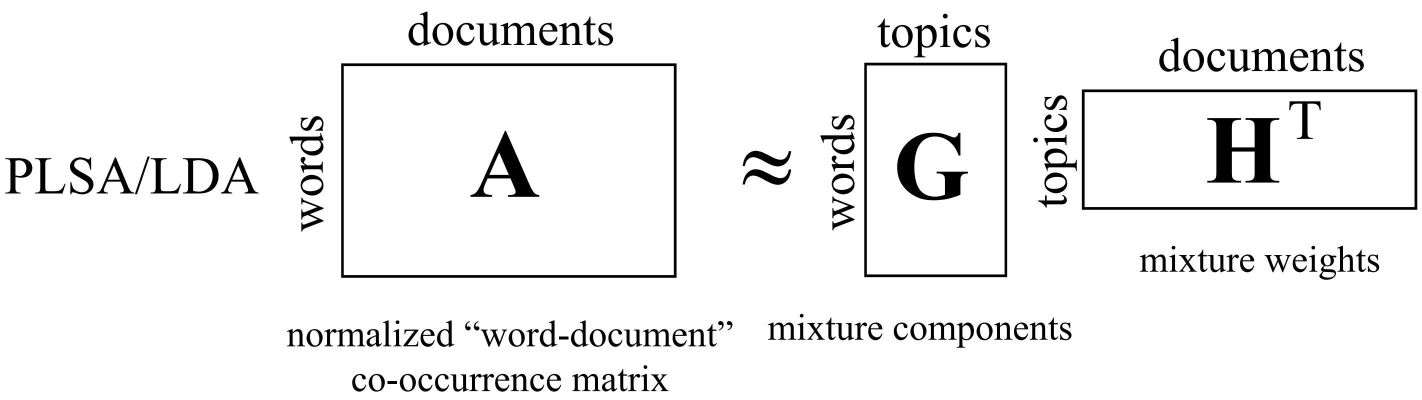
- Each word of language are treated as a word topical mixture model for predicting the occurrences of other words

$$P_{\mathrm{WTMM}}\left(w_i \mid \mathrm{M}_{w_j}\right)=\sum_{k=1}^{K} P\left(w_i \mid T_k\right) P\left(T_k \mid \mathrm{M}_{w_j}\right)$$

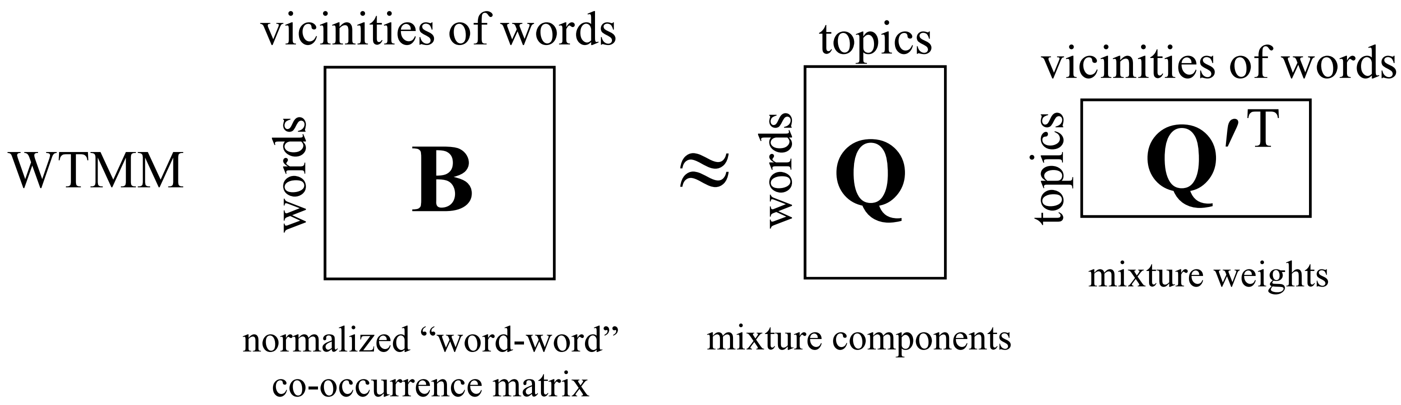
- WTMM also can be viewed as a nonnegative factorization of a “word-word” matrix consisting probability entries
  - Each column encodes the vicinity information of all occurrences of a distinct word

Comparison of WTMM and PLSA/LDA

- A schematic comparison for the matrix factorizations of PLSA/LDA and WTMM



$$P_{\mathrm{PLSA}}\left(w_i \mid \mathrm{M}_D\right) = \sum_{k=1}^{K} P\left(w_i \mid T_k\right) P\left(T_k \mid \mathrm{M}_D\right)$$



$$P_{\mathrm{WTMM}}\left(w_i \mid \mathrm{M}_{w_j}\right) = \sum_{k=1}^{K} P\left(w_i \mid T_k\right) P\left(T_k \mid \mathrm{M}_{w_j}\right)$$

WTMM: Information Retrieval (1/2)

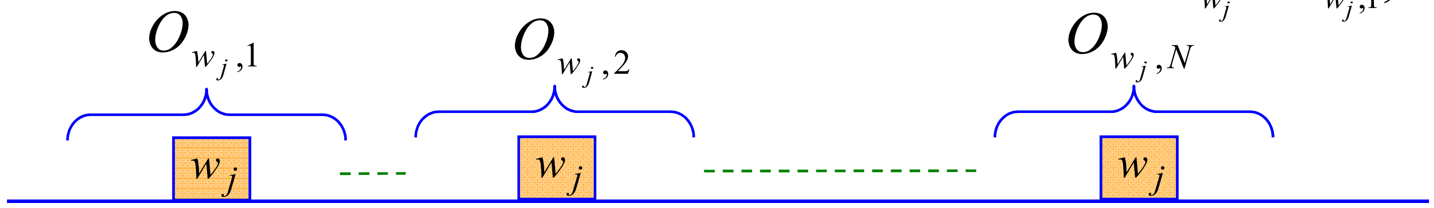
- The relevance measure between a query and a document can be expressed by

$$P_{\mathrm{WTMM}}\left(Q \middle| D\right) = \prod_{w_i \in Q} \left[\sum_{w_j \in D} \alpha_{j,D} \sum_{k=1}^{K} P\left(w_i \middle| T_k\right) P\left(T_k \middle| \mathrm{M}_{w_j}\right) \right]^{c(w_i, Q)}$$

- Unsupervised training
  - The WTMM of each word can be trained by concatenating those words occurring within a context window of size around each occurrence of the word, which are postulated to be relevant to the word

$$\log L_{\mathbf{w}} = \sum_{w_j \in \mathbf{w}} \log P_{\mathrm{WTMM}}\left(O_{w_j} \middle| \mathrm{M}_{w_j}\right) = \sum_{w_j \in \mathbf{w}} \sum_{w_i \in Q_{w_j}} c\left(w_i, O_{w_j}\right) \log P_{\mathrm{WTMM}}\left(w_i \middle| \mathrm{M}_{w_j}\right)$$

$$O_{w_j} = O_{w_j,1}, O_{w_j,2}, \cdots, O_{w_j,N}$$



WTMM: Information Retrieval (2/2)

- Supervised training: The model parameters are trained using a training set of query exemplars and the associated query-document relevance information
  - Maximize the log-likelihood of the training set of query exemplars generated by their relevant documents

$$\log L_{\mathbf{Q}_{TrainSet}} = \sum_{Q \in \mathbf{Q}_{TrainSet}} \sum_{D \in \mathbf{D}_{R \text{ to } Q}} \log P_{\text{WTMM}}(Q|D)$$

Applying Relevance Feedback to LM Framework (1/2)

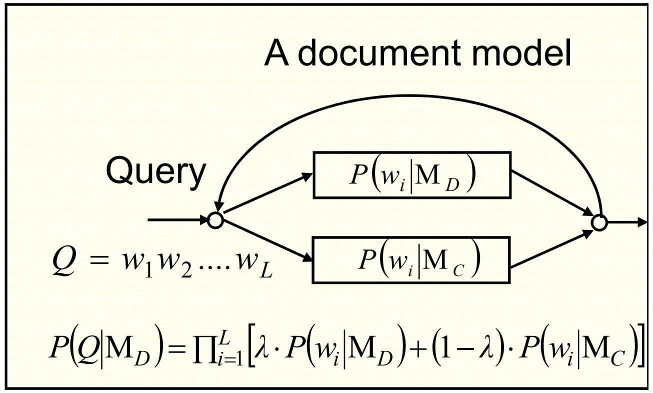
- There is still no formal mechanism to incorporate relevance feedback (judgments) into the language modeling framework
  - The query is a fixed sample while focusing on estimating accurate estimation of document language models $P(w|D)$
- Ponte (1998) proposed a limited way to incorporate blind reference feedback into the LM framework
  - Think of example relevant documents $D \in \tilde{R}$ as examples of what the query might have been, and re-sample (or expand) the query by adding *k* highly descriptive words from the these documents (blind reference feedback)

$$w^* = \arg\max_{w} \sum_{D \in \tilde{R}} \log \frac{P(w|\mathrm{M}_D)}{P(w|\mathrm{M}_C)}$$

J. M. Ponte, A language modeling approach to information retrieval, Ph.D. dissertation, UMass, 1998

Applying Relevance Feedback to LM Framework (2/2)

- Miller et al. (1999) propose two relevance feedback approach
  - Query expansion: add those words to the initial query that appear in two or more of the top m retrieved documents
  - Document model re-estimation: use a set of outside training query exemplars to train the transition probabilities of the document models



the old weight

the new weight

819 queries ≤2265 docs

$$\hat{\lambda} = \frac{\sum\limits_{Q\in[TrainSet]_Q} \sum\limits_{D\in[Doc]_{R \text{ to } Q}} \sum\limits_{q_n\in Q} \left[\frac{\lambda P(q_n|M_D)}{\lambda P(q_n|M_D)+(1-\lambda)P(q_n|M_C)}\right]}{\sum\limits_{Q\in[TrainSet]_Q} |Q|\cdot\left|[Doc]_{R \text{ to } Q}\right|}$$

- 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

Miller et al. , A hidden Markov model information retrieval system, *SIGIR 1999*

Incorporating Prior Knowledge into LM Framework

- Several efforts have been paid to using prior knowledge for the LM framework, especially modeling the document prior $P(D)$
  - Document length
  - Document source
  - Average word-length
  - Aging (time information/period)
  - URL
  - Page links

$$P(D|Q) = \frac{P(Q|D)\boxed{P(D)}}{P(Q)}$$

Implementation Notes

- For language modeling approaches to IR, many conditional probabilities are usually multiplied. This can result in a "floating point underflow"

$$P(Q|\mathrm{M}_D) = \prod_{i=1}^{L}\left[\lambda \cdot P(w_i|\mathrm{M}_D) + (1-\lambda)\cdot P(w_i|\mathrm{M}_C)\right]$$

- It is better to perform the computation by "adding" logarithms of probabilities instead
  - The logarithm function is monotonic (order-preserving)

$$\log P(Q|\mathrm{M}_D) = \sum_{i=1}^{l} \log\left[\lambda \cdot P(w_i|\mathrm{M}_D) + (1-\lambda)\cdot P(w_i|\mathrm{M}_C)\right]$$

- We also should avoid the problem of "zero probabilities (or estimates)" owing to sparse data, by using appropriate probability smoothing techniques