Chapter 4: Advanced IR Models

4.1.1 Probabilistic Retrieval: Principles
[Robertson and Sparck Jones 1976]

Goal:

Ranking based on sim(doc d, query q) =
P[R|d] = P [doc d is relevant for query q |
d has term vector X1, ..., Xm]

Assumptions:

- Relevant and irrelevant documents differ in their terms.
- Binary Independence Retrieval (BIR) Model:
  - Probabilities for term occurrence are pairwise independent for different terms.
  - Term weights are binary $\in$ {0,1}.
- For terms that do not occur in query q the probabilities for such a term occurring are the same for relevant and irrelevant documents.

4.1.2 Probabilistic IR with Term Independence: Ranking Proportional to Relevance Odds

$$sim(d,q) = O(R \mid d) = \frac{P[R \mid d]}{P[\neg R \mid d]}$$ (odds for relevance)

$$= \frac{P[d \mid R] \times P[R]}{P[d \mid \neg R] \times P[\neg R]}$$ (Bayes‘ theorem)

$$\sim \frac{P[d \mid R]}{P[d \mid \neg R]} \quad = \prod_i \frac{P[X_i \mid R]}{P[X_i \mid \neg R]}$$ (independence or linked dependence)

$$sim(d,q)' = \log \prod_{i \in q} \frac{P[Xi \mid R]}{P[Xi \mid \neg R]}$$ (Xi = 1 if d includes i-th term, 0 otherwise)

$$= \sum_{i \in q} \log P[Xi \mid R] - \log P[Xi \mid \neg R]$$

Probabilistic Retrieval: Ranking Proportional to Relevance Odds (cont.)

$$= \sum_{i \in q} log(pi^{Xi} (1-pi)^{1-Xi}) - log(qi^{Xi} (1-qi)^{1-Xi})$$ (binary features)

with estimators pi=P[Xi=1|R] and qi=P[Xi=1|¬R]

$$= \sum_{i \in q} log(\frac{pi^{Xi} (1-pi)}{(1-pi)^{Xi}}) - log(\frac{qi^{Xi} (1-qi)}{(1-qi)^{Xi}})$$

$$= \sum_{i \in q} Xi \, log \frac{pi}{1-pi} + \sum_{i \in q} Xi \; log \frac{1-qi}{qi} + \sum_{i \in q} log \frac{1-pi}{1-qi}$$

$$\sim \sum_{i \in q} Xi \, log \frac{pi}{1-pi} + \sum_{i \in q} Xi \; log \frac{1-qi}{qi} \; = \; sim(d,q)''$$

Probabilistic Retrieval: Robertson / Sparck Jones Formula

Estimate pi und qi based on training sample
(query q on small sample of corpus) or based on
intellectual assessment of first round‘s result (***relevance feedback***):

Let N be #docs in sample,
R be # relevant docs in sample
ni #docs in sample that contain term i,
ri # relevant docs in sample that contain term i

$\Rightarrow$ Estimate: $pi = \frac{ri}{R}$ $\qquad qi = \frac{ni - ri}{N - R}$

or: $pi = \frac{ri + 0.5}{R + 1}$ $\qquad qi = \frac{ni - ri + 0.5}{N - R + 1}$ **(Lidstone smoothing with λ=0.5)**

$$\Rightarrow \quad sim(d,q)'' = \sum_i Xi \log \frac{ri + 0.5}{R - ri + 0.5} + \sum_i Xi \log \frac{N - ni - R + ri + 0.5}{ni - ri + 0.5}$$

$\Rightarrow$ Weight of term i in doc d: $\log \frac{(ri + 0.5)\,(N - ni - R + ri + 0.5)}{(R - ri + 0.5)(ni - ri + 0.5)}$

Probabilistic Retrieval: tf*idf Formula

Assumptions (without training sample or relevance feedback):

- pi is the same for all i.
- Most documents are irrelevant.
- Each individual term i is infrequent.

This implies:

- $\sum_i Xi \log \frac{pi}{1-pi} = c \sum_i Xi$ with constant c
- $qi = P[Xi = 1 \mid \neg R] \approx \frac{df_i}{N}$
- $\frac{1-qi}{qi} = \frac{N - df_i}{df_i} \approx \frac{N}{df_i}$

$$\Rightarrow \quad sim(d,q)'' = \sum_i Xi \log \frac{pi}{1-pi} + \sum_i Xi \log \frac{1-qi}{qi}$$

$$\approx c \sum_i Xi + \sum_i Xi \; idf_i$$

Scalar product over the product of tf and dampend idf values for query terms

Example for Probabilistic Retrieval

Documents with relevance feedback:

q: t1 t2 t3 t4 t5 t6

	t1	t2	t3	t4	t5	t6	R
d1	1	0	1	1	0	0	1
d2	1	1	0	1	1	0	1
d3	0	0	0	1	1	0	0
d4	0	0	1	0	0	0	0
ni	2	1	2	3	2	0	
ri	2	1	1	2	1	0	
pi	5/6	1/2	1/2	5/6	1/2	1/6	
qi	1/6	1/6	1/2	1/2	1/2	1/6	

R=2, N=4

Score of new document d5 (with Lidstone smoothing with λ=0.5):

d5$\cap$q: <1 1 0 0 0 1> $\rightarrow$ sim(d5, q) = log 5 + log 1 + log 0.2 + log 5 + log 5 + log 5

$$sim(d,q)'' = \sum_i Xi \log \frac{pi}{1-pi} + \sum_i Xi \log \frac{1-qi}{qi}$$

Laplace Smoothing (with Uniform Prior)

Probabilities pi and qi for term i are estimated by MLE for binomial distribution
(repeated coin tosses for relevant docs, showing term i with pi,
Repeated coin tosses for irrelevant docs, showing term i with qi)

To avoid overfitting to feedback/training, the estimates should be smoothed (e.g. with uniform prior):

Instead of estimating pi = k/n estimate (Laplace's law of succession):

$$pi = (k+1) / (n+2)$$

or with heuristic generalization (Lidstone's law of succession):

$$pi = (k+\lambda) / (n+2\lambda) \quad \text{with } \lambda > 0 \text{ (e.g. } \lambda=0.5)$$

And for multinomial distribution (n times w-faceted dice) estimate:

$$pi = (ki + 1) / (n + w)$$

4.1.3 Probabilistic IR with Poisson Model (Okapi BM25)

Generalize term weight $w = \log \frac{p(1-q)}{q(1-p)}$

into $w = \log \frac{p_{tf}\, q_0}{q_{tf}\, p_0}$

with p_j, q_j denoting prob. that term occurs j times in rel./irrel. doc

Postulate Poisson (or Poisson-mixture) distributions:

$$p_{tf} = e^{-\lambda} \frac{\lambda^{tf}}{tf!} \qquad q_{tf} = e^{-\mu} \frac{\mu^{tf}}{tf!}$$

Okapi BM25

Approximation of Poisson model by similarly-shaped function:

$$w := \log \frac{p(1-q)}{q(1-p)} \cdot \frac{tf}{k_1 + tf}$$

finally leads to Okapi BM25 (which achieved best TREC results):

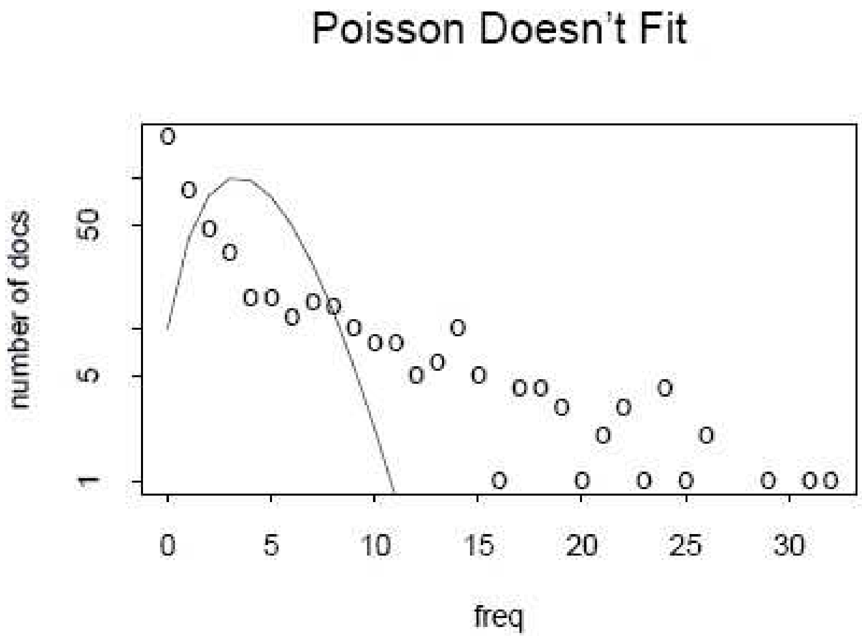
$$w_j(d) := \frac{(k_1+1)tf_j}{k_1((1-b)+b\frac{length(d)}{avgdoclength})+tf_j} \cdot \log \frac{N-df_j+0.5}{df_j+0.5}$$

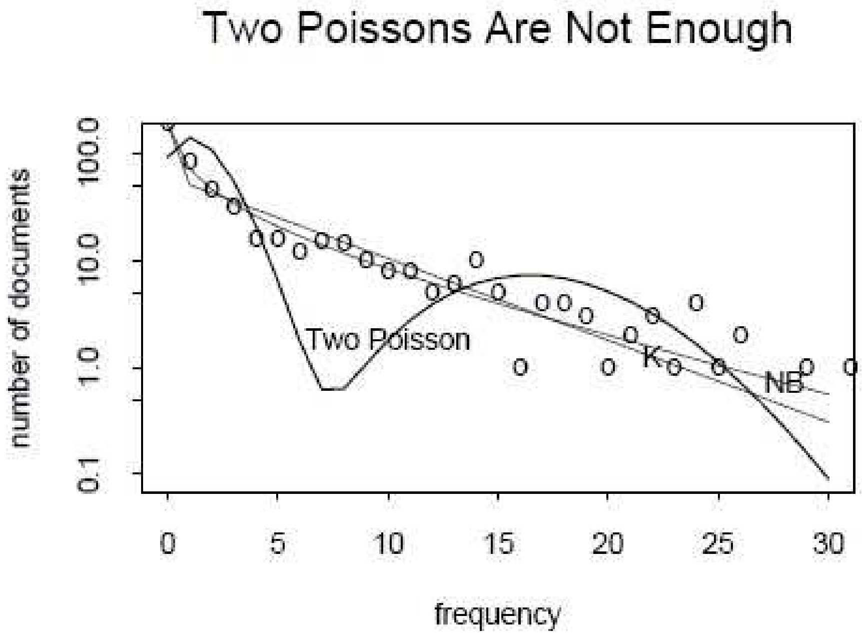
or in the most comprehensive, tunable form:

$$score(d,q) := \sum_{j=1..|q|} \log \frac{N-df_j+0.5}{df_j+0.5} \cdot \frac{(k_1+1)tf_j}{k_1((1-b)+b\frac{len(d)}{\Delta})+tf_j} \cdot \frac{(k_3+1)qtf_j}{k_3+tf_j} + k_2\,|q|\,\frac{\Delta - len(d)}{\Delta + len(d)}$$

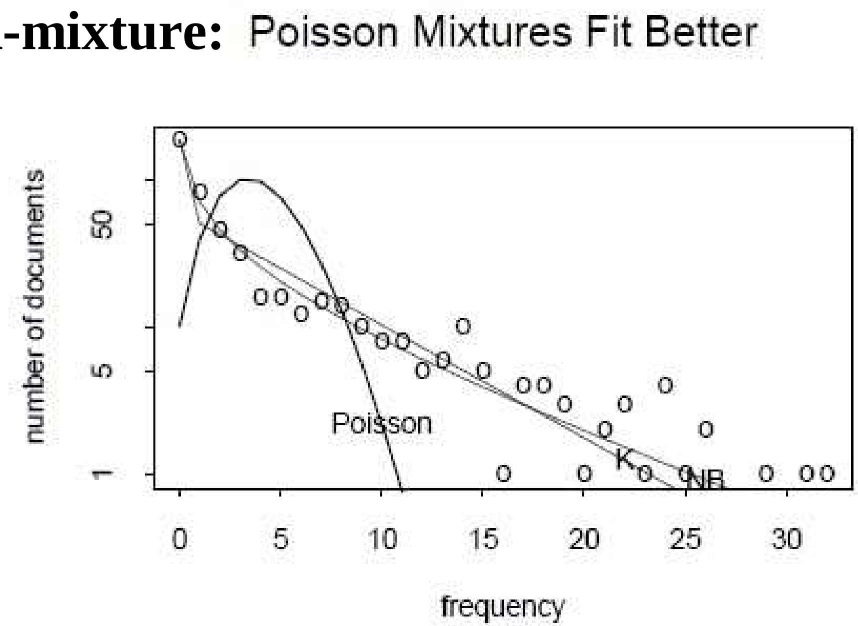
with Δ=avgdoclength and tuning parameters k_1, k_2, k_3, b, and non-linear influence of tf and consideration of doc length

Poisson Mixtures for Capturing tf Distribution





Katz's K-mixture:



distribution of tf values for term „said"

Source:
Church/Gale 1995

Katz's K-Mixture

Katz's K-mixture: $$f(k) = \int_0^\infty \Phi(\theta) \cdot \frac{e^{-\theta}\theta^k}{k!}$$

e.g. with :

$$\Phi_K(\theta) = (1-\alpha)\delta(\theta = 0) + \frac{\alpha}{\beta} e^{-\theta/\beta}$$

$$\longrightarrow f(k) = (1-\alpha)\delta(k=0) + \frac{\alpha}{\beta+1}\left(\frac{\beta}{\beta+1}\right)^k$$

with δ(G)=1 if G is true, 0 otherwise

Parameter estimation for given term:

$$\lambda = cf\ / N$$ observed mean tf

$$idf = \log_2(N / df)$$

$$\beta = \lambda 2^{idf} - 1 = (cf - df)/df$$ extra occurrences (tf>1)

$$\alpha = \lambda / \beta$$

4.1.4 Extensions of Probabilistic IR

Consider term correlations in documents (with binary Xi)
$\rightarrow$ Problem of estimating m-dimensional prob. distribution
$P[X1=... \wedge X2= ... \wedge ... \wedge Xm=...] =: f_X(X1, ..., Xm)$

One possible approach: **Tree Dependence Model**:

a) Consider only 2-dimensional probabilities (for term pairs)

$f_{ij}(Xi, Xj)=P[Xi=..\wedge Xj=..]=\sum_{X_1}..\sum_{X_{i-1}}\sum_{X_{i+1}}..\sum_{X_{j-1}}\sum_{X_{j+1}}..\sum_{X_m} P[X_1 = ... \wedge .. \wedge X_m = ...]$

b) For each term pair

estimate the error between independence and the actual correlation

c) Construct a tree with terms as nodes and the

m-1 highest error (or correlation) values as weighted edges

Considering Two-dimensional Term Correlation

Variant 1:

Error of approximating f by g (**Kullback-Leibler divergence**) with g assuming pairwise term independence:

$$\varepsilon(f,g) := \sum_{\vec{X} \in \{0,1\}^m} f(\vec{X}) \log \frac{f(\vec{X})}{g(\vec{X})} = \sum_{\vec{X} \in \{0,1\}^m} f(\vec{X}) \log \frac{f(\vec{X})}{\prod_{i=1}^{m} g_i(X_i)}$$

Variant 2:

Correlation coefficient for term pairs:

$$\rho(Xi, Xj) := \frac{Cov(Xi, Xj)}{\sqrt{Var(Xi)}\sqrt{Var(Xj)}}$$

Variant 3:

level-α values or p-values
of **Chi-square independence test**

Example for Approximation Error ε (KL Strength)

m=2:

given are documents:

d1=(1,1), d2(0,0), d3=(1,1), d4=(0,1)

estimation of 2-dimensional prob. distribution f:

$f(1,1) = P[X1=1 \wedge X2=1] = 2/4$

$f(0,0) = 1/4$, $f(0,1) = 1/4$, $f(1,0) = 0$

estimation of 1-dimensional marginal distributions g1 and g2:

$g1(1) = P[X1=1] = 2/4$, $g1(0) = 2/4$

$g2(1) = P[X2=1] = 3/4$, $g2(0) = 1/4$

estimation of 2-dim. distribution g with independent Xi:

$g(1,1) = g1(1)*g2(1) = 3/8$,

$g(0,0) = 1/8$, $g(0,1) = 3/8$, $g(1,0) =1/8$

approximation error ε (KL divergence):

$$\varepsilon = 2/4 \log 4/3 \; + \; 1/4 \log 2 \; + \; 1/4 \log 2/3 \; + 0$$

Constructing the Term Dependence Tree

<u>Given:</u>

complete graph (V, E) with m nodes $X_i \in V$ and
m^2 undirected edges $\in E$ with weights ε (or ρ)

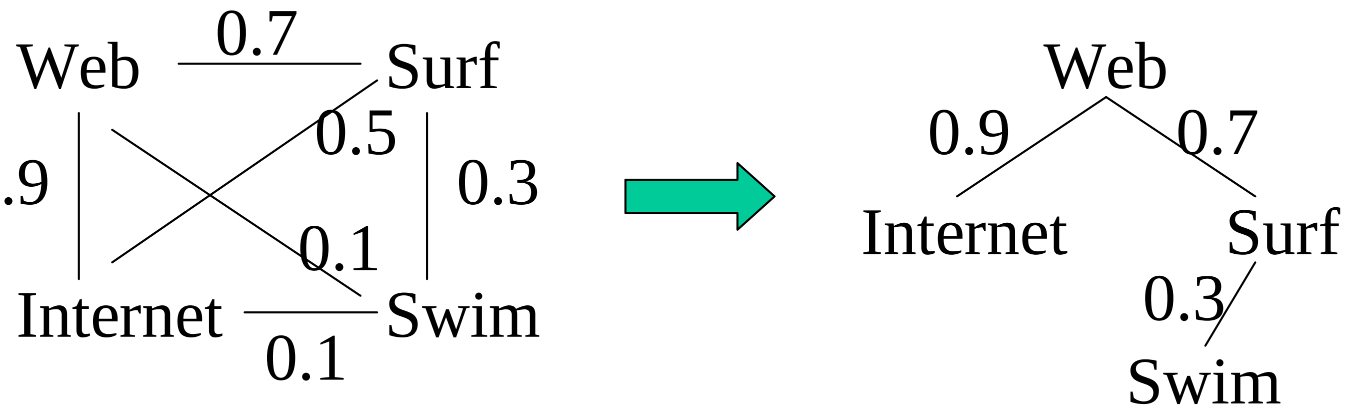
<u>Wanted:</u>

spanning tree (V, E‘) with maximal sum of weights

<u>Algorithm:</u>

Sort the m^2 edges of E in descending order of weight
$E' := \emptyset$
Repeat until $|E'| = m-1$
  $E' := E' \cup \{(i,j) \in E \mid (i,j)$ has max. weight in $E\}$
  provided that E‘ remains acyclic;
  $E := E - \{(i,j) \in E \mid (i,j)$ has max. weight in $E\}$

Example:



Estimation of Multidimensional Probabilities with Term Dependence Tree

Given is a term dependence tree (V = {X1, ..., Xm}, E‘).
Let X1 be the root, nodes are preorder-numbered, and assume that Xi and Xj are independent for (i,j) $\notin$ E‘. Then:

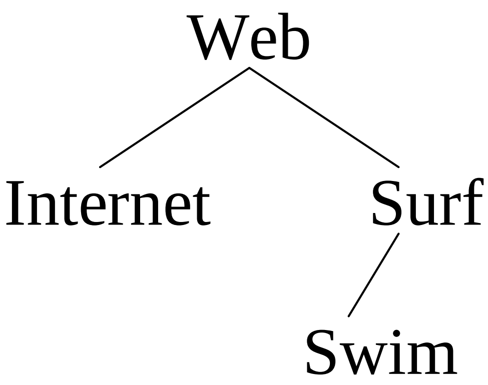
$$P[X1=..\wedge..\wedge Xm=..] = P[X1=..]\, P[X2=..\wedge Xm=..\,|\, X1=..]$$

$$=\Pi_{i=1..m} P[Xi=..\,|\, X1=..\wedge X(i-1)=..]$$

$$=P[X1]\cdot \prod_{(i,j)\in E'} P[Xj\,|\,Xi]$$

$$=P[X1]\cdot \prod_{(i,j)\in E'} \frac{P[Xi,Xj]}{P[Xi]}$$

<u>Example:</u>



P[Web, Internet, Surf, Swim] =

$$P[Web]\frac{P[Web,Internet]}{P[Web]}\frac{P[Web,Surf]}{P[Web]}\frac{P[Surf,Swim]}{P[Surf]}$$

Bayesian Networks

A **Bayesian network (BN) is a directed, acyclic graph (V, E)** with the following properties:

- Nodes $\in$ V representing random variables and
- Edges $\in$ E representing dependencies.
- For a root R $\in$ V the BN captures the prior probability P[R = ...].
- For a node X $\in$ V with parents parents(X) = {P1, ..., Pk} the BN captures the conditional probability P[X=... | P1, ..., Pk].
- Node X is conditionally independent of a non-parent node Y given its parents parents(X) = {P1, ..., Pk}: P[X | P1, ..., Pk, Y] = P[X | P1, ..., Pk].

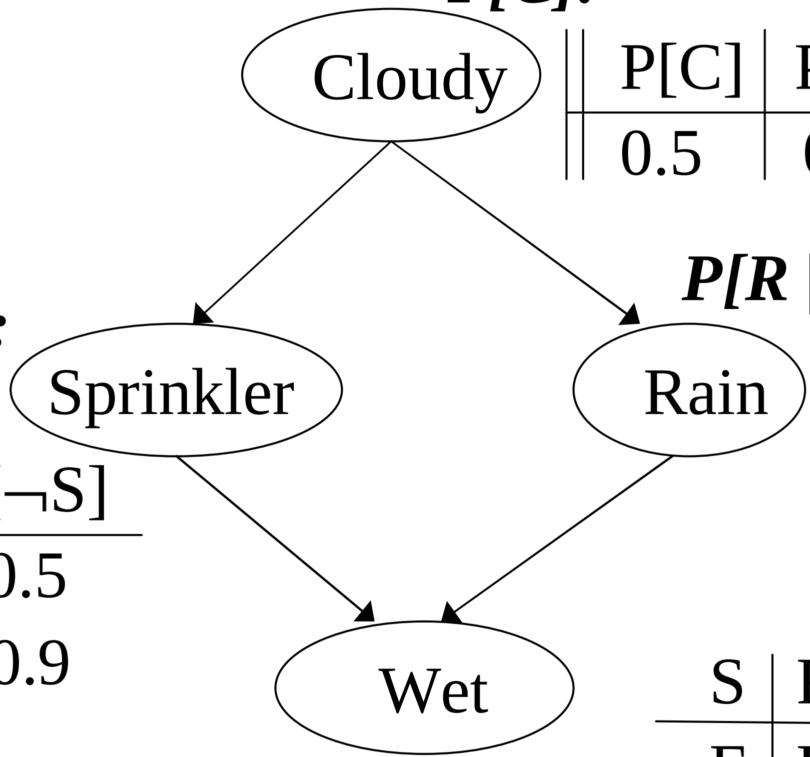
This implies: $P[X1 \ldots Xn] = P[X1 | X2 \ldots Xn]\, P[X2 \ldots Xn]$

- by the chain rule: $= \prod_{i=1}^{n} P[Xi \,|\, X(i+1) \ldots Xn]$

- by cond. independence: $= \prod_{i=1}^{n} P[Xi \,|\, parents(Xi), other\ \ nodes]$

$$= \prod_{i=1}^{n} P[Xi \,|\, parents(Xi)]$$

Example of Bayesian Network (Belief Network)



P[C]:

P[C]	P[¬C]
0.5	0.5

P[S | C]:

C	P[S]	P[¬S]
F	0.5	0.5
T	0.1	0.9

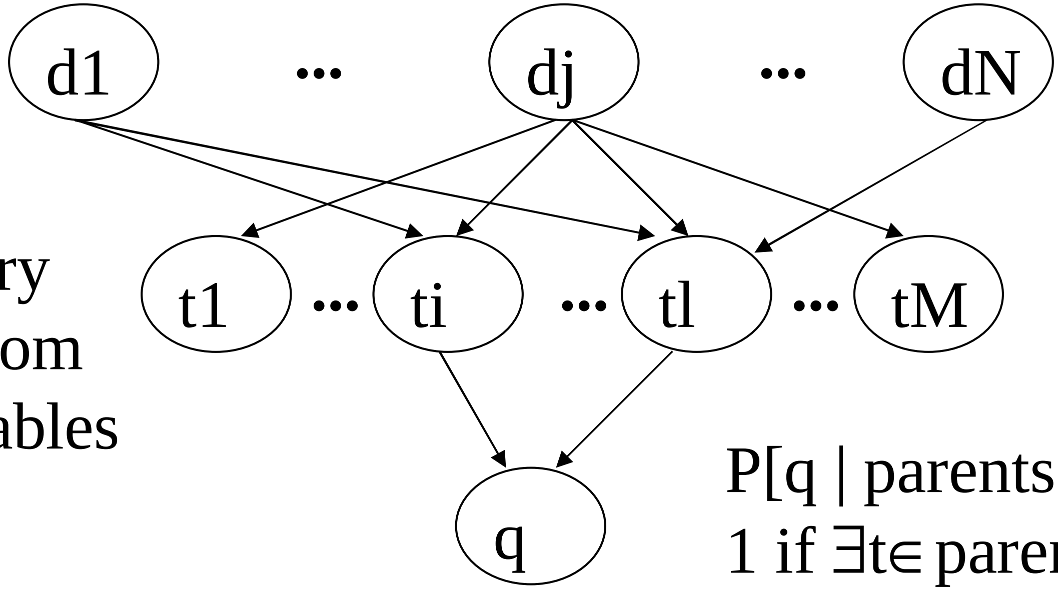
P[R | C]:

C	P[R]	P[¬R]
F	0.2	0.8
T	0.8	0.2

P[W | S,R]:

S	R	P[W]	P[¬W]
F	F	0.0	1.0
F	T	0.9	0.1
T	F	0.9	0.1
T	T	0.99	0.01

Bayesian Inference Networks for IR



with binary random variables

P[dj]=1/N

P[ti | dj∈parents(ti)] =
1 if ti occurs in dj,
0 otherwise

P[q | parents(q)] =
1 if ∃t∈parents(q): t is relevant for q,
0 otherwise

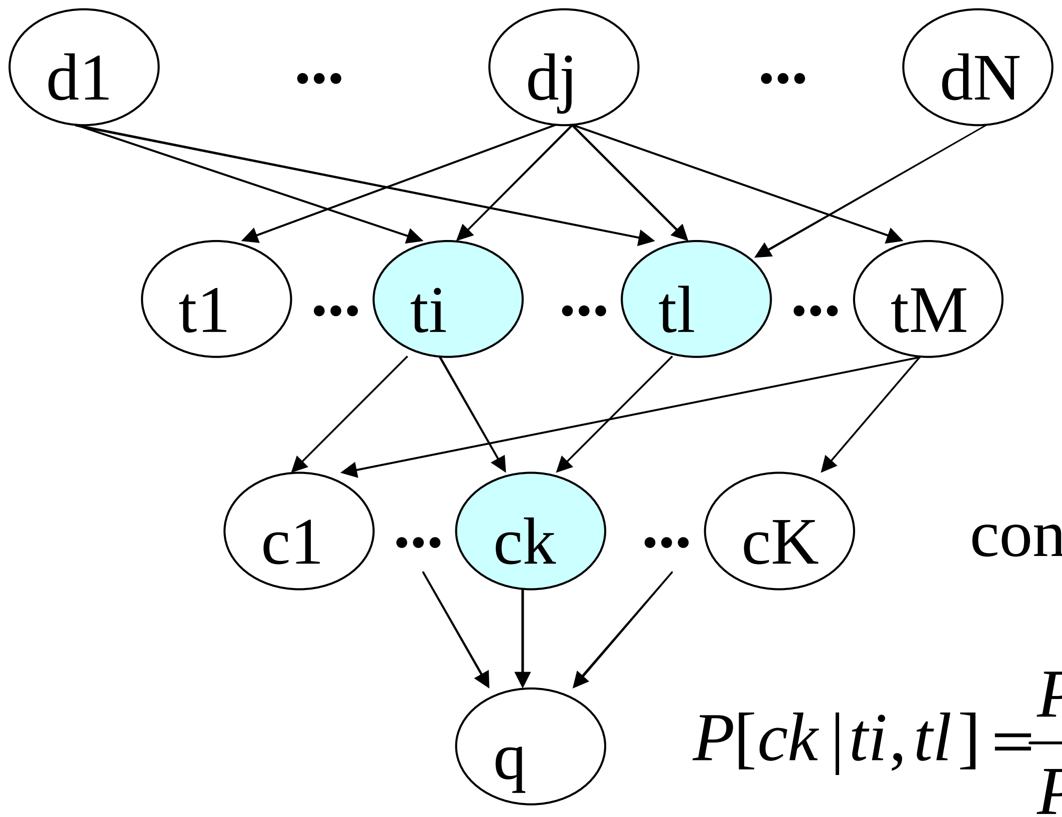
$$P[q \wedge dj] = \sum_{(t1 \ldots tM)} P[q \wedge dj | t1 \ldots tM]\, P[t1 \ldots tM]$$

$$= \sum_{(t1 \ldots tM)} P[q \wedge dj \wedge t1 \wedge \ldots \wedge tM]$$

$$= \sum_{(t1 \ldots tM)} P[q | dj \wedge t1 \wedge \ldots \wedge tM]\, P[dj \wedge t1 \wedge \ldots \wedge tM]$$

$$= \sum_{(t1 \ldots tM)} P[q | t1 \wedge \ldots \wedge tM]\, P[t1 \wedge \ldots \wedge tM | dj]\, P[dj]$$

Advanced Bayesian Network for IR



concepts / topics

$$P[ck \mid ti, tl] = \frac{P[ti \wedge tl]}{P[ti \vee tl]} \approx \frac{df_{il}}{df_i + df_l - df_{il}}$$

Problems:

- parameter estimation (sampling / training)
- (non-) scalable representation
- (in-) efficient prediction
- fully convincing experiments

Additional Literature for Chapter 4

Probabilistic IR:

- Grossman/Frieder Sections 2.2 and 2.4
- S.E. Robertson, K. Sparck Jones: Relevance Weighting of Search Terms, JASIS 27(3), 1976
- S.E. Robertson, S. Walker: Some Simple Effective Approximations to the 2-Poisson Model for Probabilistic Weighted Retrieval, SIGIR 1994
  K.W. Church, W.A. Gale: Poisson Mixtures, Natural Language Engineering 1(2), 1995
- C.T. Yu, W. Meng: Principles of Database Query Processing for Advanced Applications, Morgan Kaufmann, 1997, Chapter 9
- D. Heckerman: A Tutorial on Learning with Bayesian Networks, Technical Report MSR-TR-95-06, Microsoft Research, 1995