CS276A
Text Retrieval and Mining

Lecture 10

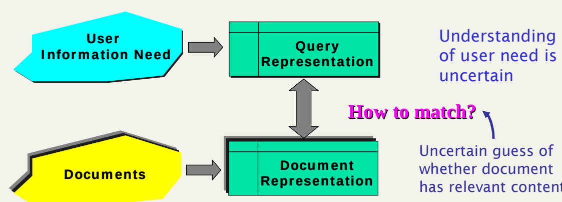
Recap of the last lecture

- Improving search results
  - Especially for high recall. E.g., searching for *aircraft* so it matches with *plane; thermodynamic* with *heat*
- Options for improving results…
  - Global methods
    - Query expansion
      - Thesauri
      - Automatic thesaurus generation
    - Global indirect relevance feedback
  - Local methods
    - Relevance feedback
    - Pseudo relevance feedback

Probabilistic relevance feedback

- Rather than reweighting in a vector space…
- If user has told us some relevant and some irrelevant documents, then we can proceed to build a probabilistic classifier, such as a Naive Bayes model:
  - $P(t_k|R) = |\mathbf{D}_{rk}| / |\mathbf{D}_r|$
  - $P(t_k|NR) = |\mathbf{D}_{nrk}| / |\mathbf{D}_{nr}|$
    - t_k is a term; $\mathbf{D}_r$ is the set of known relevant documents; $\mathbf{D}_{rk}$ is the subset that contain t_k; $\mathbf{D}_{nr}$ is the set of known irrelevant documents; $\mathbf{D}_{nrk}$ is the subset that contain t_k.

Why probabilities in IR?



In traditional IR systems, matching between each document and query is attempted in a semantically imprecise space of index terms.

Probabilities provide a principled foundation for uncertain reasoning. *Can we use probabilities to quantify our uncertainties?*

Probabilistic IR topics

- Classical probabilistic retrieval model
  - Probability ranking principle, etc.
- (Naïve) Bayesian Text Categorization
- Bayesian networks for text retrieval
- Language model approach to IR
  - An important emphasis in recent work

- *Probabilistic methods are one of the oldest but also one of the currently hottest topics in IR.*
  - *Traditionally: neat ideas, but they've never won on performance. It may be different now.*

The document ranking problem

- We have a collection of documents
- User issues a query
- A list of documents needs to be returned
- **Ranking method is core of an IR system:**
  - **In what order do we present documents to the user?**
  - We want the "best" document to be first, second best second, etc….
- **Idea: Rank by probability of relevance of the document w.r.t. information need**
  - P(relevant|document$_i$, query)

Recall a few probability basics

- For events *a* and *b:*
- Bayes' Rule

$$p(a,b) = p(a \cap b) = p(a \mid b)\,p(b) = p(b \mid a)\,p(a)$$

$$p(\bar{a} \mid b)\,p(b) = p(b \mid \bar{a})\,p(\bar{a})$$

$$p(a \mid b) = \frac{p(b \mid a)\,p(a)}{p(b)} = \frac{p(b \mid a)\,p(a)}{\sum_{x=a,\bar{a}} p(b \mid x)\,p(x)}$$

Prior

Posterior

- Odds: $O(a) = \dfrac{p(a)}{p(\bar{a})} = \dfrac{p(a)}{1 - p(a)}$

The Probability Ranking Principle

"If a reference retrieval system's response to each request is a ranking of the documents in the collection in order of decreasing probability of relevance to the user who submitted the request, where the probabilities are estimated as accurately as possible on the basis of whatever data have been made available to the system for this purpose, the overall effectiveness of the system to its user will be the best that is obtainable on the basis of those data."

- [1960s/1970s] S. Robertson, W.S. Cooper, M.E. Maron; van Rijsbergen (1979:113); Manning & Schütze (1999:538)

Probability Ranking Principle

Let x be a document in the collection.
Let R represent **relevance** of a document w.r.t. given (fixed) query and let NR represent **non-relevance.** R={0,1} vs. NR/R

Need to find $p(R|x)$ - probability that a document x is **relevant.**

$$p(R \mid x) = \frac{p(x \mid R)\,p(R)}{p(x)}$$

$$p(NR \mid x) = \frac{p(x \mid NR)\,p(NR)}{p(x)}$$

$p(R)$,$p(NR)$ - prior probability of retrieving a (non) relevant document

$$p(R \mid x) + p(NR \mid x) = 1$$

$p(x|R)$, $p(x|NR)$ - probability that if a relevant (non-relevant) document is retrieved, it is x.

Probability Ranking Principle (PRP)

- Simple case: no selection costs or other utility concerns that would differentially weight errors
- ***Bayes' Optimal Decision Rule***
  - ***x*** is **relevant** <u>iff</u> $p(R|x) > p(NR|x)$
- PRP in action: Rank all documents by $p(R|x)$
- Theorem:
  - Using the PRP is optimal, in that it minimizes the loss (Bayes risk) under 1/0 loss
  - Provable if all probabilities correct, etc. [e.g., Ripley 1996]

Probability Ranking Principle

- More complex case: retrieval costs.
  - Let *d* be a document
  - *C* - cost of retrieval of <u>relevant</u> document
  - *C'* - cost of retrieval of <u>non-relevant</u> document
- Probability Ranking Principle: if

$$C \cdot p(R \mid d) + C' \cdot (1 - p(R \mid d)) \le C \cdot p(R \mid d') + C' \cdot (1 - p(R \mid d'))$$

for all *d' not yet retrieved*, then *d* **is the next document to be retrieved**

- **We won't further consider loss/utility from now on**

Probability Ranking Principle

- How do we compute all those probabilities?
  - Do not know exact probabilities, have to use estimates
  - Binary Independence Retrieval (BIR) – which we discuss later today – is the simplest model
- Questionable assumptions
  - "Relevance" of each document is independent of relevance of other documents.
    - Really, it's bad to keep on returning **duplicates**
  - Boolean model of relevance
  - That one has a single step information need
    - Seeing a range of results might let user refine query

Probabilistic Retrieval Strategy

- Estimate how terms contribute to relevance
  - How do things like tf, df, and length influence your judgments about document relevance?
    - One answer is the Okapi formulae (S. Robertson)
- Combine to find document relevance probability
- Order documents by decreasing probability

Probabilistic Ranking

Basic concept:

"For a given query, if we know some documents that are relevant, terms that occur in those documents should be given greater weighting in searching for other relevant documents.

By making assumptions about the distribution of terms and applying Bayes Theorem, it is possible to derive weights theoretically."

Van Rijsbergen

Binary Independence Model

- Traditionally used in conjunction with PRP
- **"Binary" = Boolean**: documents are represented as binary incidence vectors of terms (cf. lecture 1):
  - $\vec{x} = (x_1, \ldots, x_n)$
  - $x_i = 1$ iff term *i* is present in document *x*.
- **"Independence":** terms occur in documents independently
- Different documents can be modeled as same vector
- Bernoulli Naive Bayes model (cf. text categorization!)

Binary Independence Model

- Queries: binary term incidence vectors
- Given query ***q***,
  - for each document ***d*** need to compute ***p(R|q,d)***.
  - replace with computing ***p(R|q,x)*** where ***x*** is binary term incidence vector representing ***d*** Interested only in ranking
- Will use odds and Bayes' Rule:

$$O(R \mid q, \vec{x}) = \frac{p(R \mid q, \vec{x})}{p(NR \mid q, \vec{x})} = \frac{\dfrac{p(R \mid q)\, p(\vec{x} \mid R, q)}{p(\vec{x} \mid q)}}{\dfrac{p(NR \mid q)\, p(\vec{x} \mid NR, q)}{p(\vec{x} \mid q)}}$$

Binary Independence Model

$$O(R \mid q, \vec{x}) = \frac{p(R \mid q, \vec{x})}{p(NR \mid q, \vec{x})} = \underbrace{\frac{p(R \mid q)}{p(NR \mid q)}}_{\text{Constant for a given query}} \cdot \underbrace{\frac{p(\vec{x} \mid R, q)}{p(\vec{x} \mid NR, q)}}_{\text{Needs estimation}}$$

- Using **Independence** Assumption:

$$\frac{p(\vec{x} \mid R, q)}{p(\vec{x} \mid NR, q)} = \prod_{i=1}^{n} \frac{p(x_i \mid R, q)}{p(x_i \mid NR, q)}$$

- So : $O(R \mid q, d) = O(R \mid q) \cdot \prod_{i=1}^{n} \frac{p(x_i \mid R, q)}{p(x_i \mid NR, q)}$

Binary Independence Model

$$O(R \mid q, d) = O(R \mid q) \cdot \prod_{i=1}^{n} \frac{p(x_i \mid R, q)}{p(x_i \mid NR, q)}$$

- Since x_i is either 0 or 1:

$$O(R \mid q, d) = O(R \mid q) \cdot \prod_{x_i = 1} \frac{p(x_i = 1 \mid R, q)}{p(x_i = 1 \mid NR, q)} \cdot \prod_{x_i = 0} \frac{p(x_i = 0 \mid R, q)}{p(x_i = 0 \mid NR, q)}$$

- Let $p_i = p(x_i = 1 \mid R, q)$; $r_i = p(x_i = 1 \mid NR, q)$;
- Assume, for all terms not occurring in the query ($q_i = 0$) $p_i = r_i$

This can be changed (e.g., in relevance feedback)

Then...

Binary Independence Model

$$O(R|q,\vec{x}) = O(R|q)\cdot \prod_{x_i=q_i=1}\frac{p_i}{r_i}\cdot\prod_{\substack{x_i=0\\ q_i=1}}\frac{1-p_i}{1-r_i}$$

All matching terms

Non-matching query terms

$$= O(R|q)\cdot \prod_{x_i=q_i=1}\frac{p_i(1-r_i)}{r_i(1-p_i)}\cdot\prod_{q_i=1}\frac{1-p_i}{1-r_i}$$

All matching terms

All query terms

Binary Independence Model

$$O(R|q,\vec{x}) = O(R|q)\cdot \prod_{x_i=q_i=1}\frac{p_i(1-r_i)}{r_i(1-p_i)}\cdot\prod_{q_i=1}\frac{1-p_i}{1-r_i}$$

Constant for each query

Only quantity to be estimated for rankings

- Retrieval Status Value:

$$RSV = \log\prod_{x_i=q_i=1}\frac{p_i(1-r_i)}{r_i(1-p_i)} = \sum_{x_i=q_i=1}\log\frac{p_i(1-r_i)}{r_i(1-p_i)}$$

Binary Independence Model

- All boils down to computing RSV.

$$RSV = \log\prod_{x_i=q_i=1}\frac{p_i(1-r_i)}{r_i(1-p_i)} = \sum_{x_i=q_i=1}\log\frac{p_i(1-r_i)}{r_i(1-p_i)}$$

$$RSV = \sum_{x_i=q_i=1} c_i; \qquad c_i = \log\frac{p_i(1-r_i)}{r_i(1-p_i)}$$

So, how do we compute c_i's from our data ?

Binary Independence Model

- Estimating RSV coefficients.
- For each term i look at this table of document counts:

Documens	Relevant	Non-Relevant	Total
X_i=1	*s*	*n-s*	*n*
X_i=0	*S-s*	*N-n-S+s*	*N-n*
Total	*S*	*N-S*	*N*

- Estimates: $p_i \approx \frac{s}{S} \quad r_i \approx \frac{(n-s)}{(N-S)}$

$$c_i \approx K(N,n,S,s) = \log\frac{s/(S-s)}{(n-s)/(N-n-S+s)}$$

For now, assume no zero terms. More next lecture.

Estimation – key challenge

- If non-relevant documents are approximated by the whole collection, then r_i (prob. of occurrence in non-relevant documents for query) *is n/N* and
  - log $(1-r_i)/r_i$ = log (N– *n*)/*n* ≈ log N/*n* = IDF!
- p_i (probability of occurrence in relevant documents) can be estimated in various ways:
  - from relevant documents if know some
    - Relevance weighting can be used in feedback loop
  - constant (Croft and Harper combination match) – then just get idf weighting of terms
  - proportional to prob. of occurrence in collection
    - more accurately, to log of this (Greiff, SIGIR 1998)

Iteratively estimating p_i

1. Assume that p_i constant over all x_i in query
   - p_i = 0.5 (even odds) for any given doc
2. Determine guess of relevant document set:
   - V is fixed size set of highest ranked documents on this model (note: now a bit like tf.idf!)
3. We need to improve our guesses for p_i and r_i, so
   - Use distribution of x_i in docs in V. Let V_i be set of documents containing x_i
     - $p_i = |V_i| / |V|$
   - Assume if not retrieved then not relevant
     - $r_i = (n_i - |V_i|) / (N - |V|)$
4. Go to 2. until converges then return ranking

24

Probabilistic Relevance Feedback

1. Guess a preliminary probabilistic description of *R* and use it to retrieve a first set of documents V, as above.
2. Interact with the user to refine the description: learn some definite members of R and NR
3. Reestimate p_i and r_i on the basis of these
   - Or can combine new information with original guess (use Bayesian prior): $p_i^{(2)} = \frac{|V_i| + \kappa p_i^{(1)}}{|V| + \kappa}$

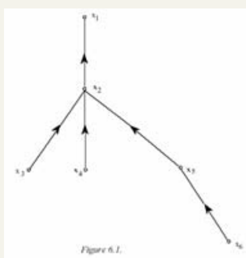
   κ is prior weight
4. Repeat, thus generating a succession of approximations to *R*.

PRP and BIR

- Getting reasonable approximations of probabilities is possible.
- Requires restrictive assumptions:
  - ***term independence***
  - ***terms not in query don't affect the outcome***
  - ***boolean representation of documents/queries/relevance***
  - ***document relevance values are independent***
- Some of these assumptions can be removed
- Problem: either require partial relevance information or only can derive somewhat inferior term weights

Removing term independence

- In general, index terms aren't independent
- Dependencies can be complex
- van Rijsbergen (1979) proposed model of simple tree dependencies
  - Exactly Friedman and Goldszmidt's Tree Augmented Naive Bayes (AAAI 13, 1996)
- Each term dependent on one other
- In 1970s, estimation problems held back success of this model



Food for thought

- Think through the differences between standard tf.idf and the probabilistic retrieval model in the first iteration
- Think through the differences between vector space (pseudo) relevance feedback and probabilistic (pseudo) relevance feedback

Good and Bad News

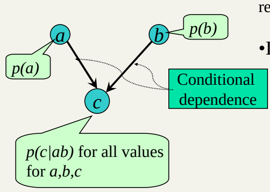
- Standard Vector Space Model
  - Empirical for the most part; success measured by results
  - Few properties provable
- Probabilistic Model Advantages
  - Based on a firm theoretical foundation
  - Theoretically justified optimal ranking scheme
- Disadvantages
  - Making the initial guess to get V
  - Binary word-in-doc weights (not using term frequencies)
  - Independence of terms (can be alleviated)
  - Amount of computation
  - Has never worked convincingly better in practice

Bayesian Networks for Text Retrieval (Turtle and Croft 1990)

- Standard probabilistic model assumes you can't estimate P(R|D,Q)
  - Instead assume independence and use P(D|R)
- But maybe you can with a Bayesian network*
- What is a Bayesian network?
  - A directed acyclic graph
  - Nodes
    - Events or Variables
      - Assume values.
      - For our purposes, all Boolean
  - Links
    - model direct dependencies between nodes

Bayesian Networks

a,b,c - propositions (events).



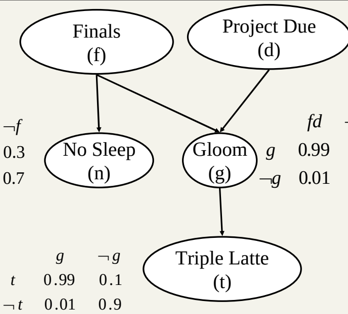
- Bayesian networks model causal relations between events
- Inference in Bayesian Nets:
  - Given probability distributions for roots and conditional probabilities can compute apriori *probability* of any instance
  - Fixing assumptions (e.g., *b* was observed) will cause recomputation of probabilities

For more information see:
R.G. Cowell, A.P. Dawid, S.L. Lauritzen, and D.J. Spiegelhalter. 1999. ***Probabilistic Networks and Expert Systems***. Springer Verlag.
J. Pearl. 1988. ***Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.*** Morgan-Kaufman.

Toy Example

f	0.3
¬*f*	0.7

d	0.4
¬*d*	0.6

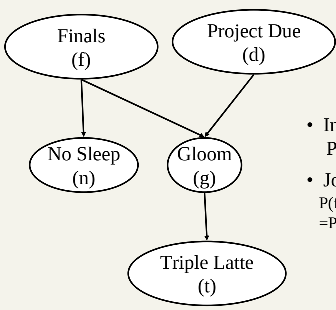


	f	¬*f*
n	0.9	0.3
¬*n*	0.1	0.7

	fd	¬*fd*	*f*¬*d*	¬*f*¬*d*
g	0.99	0.9	0.8	0.3
¬*g*	0.01	0.1	0.2	0.7

	g	¬*g*
t	0.99	0.1
¬*t*	0.01	0.9

Independence Assumptions



- Independence assumption: P(t|g, f)=P(t|g)
- Joint probability
  P(f d n g t)
  =P(f) P(d) P(n|f) P(g|f d) P(t|g)

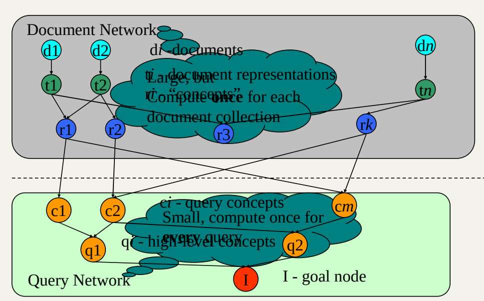
Chained inference

- Evidence - a node takes on some value
- Inference
  - Compute *belief* (probabilities) of other nodes
    - conditioned on the known evidence
  - Two kinds of inference: *Diagnostic* and *Predictive*
- Computational complexity
  - General network: NP-hard
    - Tree-like networks are easily tractable
    - Much other work on efficient exact and approximate Bayesian network inference
      - Clever dynamic programming
      - Approximate inference (“loopy belief propagation”)

Model for Text Retrieval

- Goal
  - Given a user’s information need (evidence), find probability a doc satisfies need
- Retrieval model
  - Model docs in a *document network*
  - Model information need in a *query network*

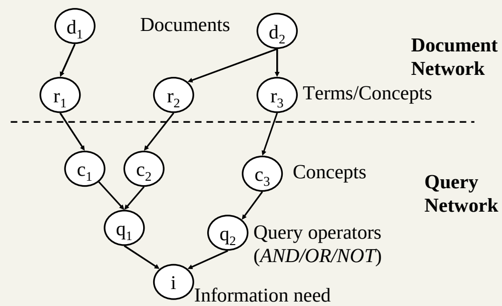
Bayesian Nets for IR: Idea



Bayesian Nets for IR

- Construct Document Network (once !)
- For each query
  - Construct best Query Network
  - Attach it to Document Network
  - Find subset of **d_i's** which maximizes the probability value of node **I** (best subset).
  - Retrieve these **d_i's** as the answer to query.

Bayesian nets for text retrieval

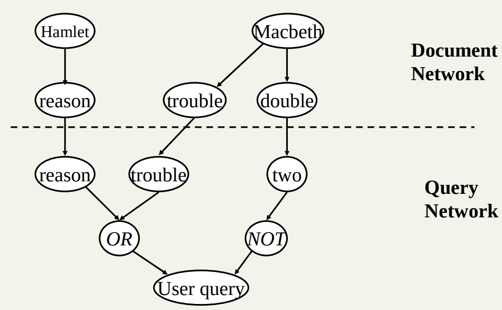


Link matrices and probabilities

- Prior doc probability *P*(*d*) = 1/*n*
- *P*(*r*|*d*)
  - within-document term frequency
  - *tf* × *idf* - based
- *P*(*c*|*r*)
  - 1-to-1
  - thesaurus
- *P*(*q*|*c*): canonical forms of query operators
  - Always use things like AND and NOT – never store a full CPT*

*conditional probability table

Example: *"reason trouble –two"*



Extensions

- Prior probs don't have to be *1/n*.
- "User information need" doesn't have to be a query - can be words typed, in docs read, any combination …
- Phrases, inter-document links
- Link matrices can be modified over time.
  - User feedback.
  - The promise of "personalization"

Computational details

- Document network built at indexing time
- Query network built/scored at query time
- Representation:
  - Link matrices from docs to any single term are like the postings entry for that term
  - Canonical link matrices are efficient to store and compute
- Attach evidence only at roots of network
  - Can do single pass from roots to leaves

Bayes Nets in IR

- Flexible ways of combining term weights, which can generalize previous approaches
  - Boolean model
  - Binary independence model
  - Probabilistic models with weaker assumptions
- Efficient large-scale implementation
  - InQuery text retrieval system from U Mass
    - Turtle and Croft (1990) [Commercial version defunct?]
- Need approximations to avoid intractable inference
- Need to estimate all the probabilities by some means (whether more or less ad hoc)
- Much new Bayes net technology yet to be applied?

Resources

S. E. Robertson and K. Spärck Jones. 1976. Relevance Weighting of Search Terms. *Journal of the American Society for Information Sciences* 27(3): 129–146.

C. J. van Rijsbergen. 1979. *Information Retrieval.* 2nd ed. London: Butterworths, chapter 6. [Most details of math] http://www.dcs.gla.ac.uk/Keith/Preface.html

N. Fuhr. 1992. Probabilistic Models in Information Retrieval. *The Computer Journal*, 35(3),243–255. [Easiest read, with BNs]

F. Crestani, M. Lalmas, C. J. van Rijsbergen, and I. Campbell. 1998. Is This Document Relevant? ... Probably: A Survey of Probabilistic Models in Information Retrieval. *ACM Computing Surveys* 30(4): 528–552.

http://www.acm.org/pubs/citations/journals/surveys/1998-30-4/p528-crestani/

[Adds very little material that isn't in van Rijsbergen or Fuhr]

Resources

H.R. Turtle and W.B. Croft. 1990. Inference Networks for Document Retrieval. *Proc. ACM SIGIR*: 1-24.

E. Charniak. Bayesian nets without tears. *AI Magazine* 12(4): 50-63 (1991). *http://www.aaai.org/Library/Magazine/Vol12/12-04/vol12-04.html*

D. Heckerman. 1995. A Tutorial on Learning with Bayesian Networks. Microsoft Technical Report MSR-TR-95-06 *http://www.research.microsoft.com/~heckerman/*

N. Fuhr. 2000. Probabilistic Datalog: Implementing Logical Information Retrieval for Advanced Applications. *Journal of the American Society for Information Science* 51(2): 95–110.

R. K. Belew. 2001. *Finding Out About: A Cognitive Perspective on Search Engine Technology and the WWW*. Cambridge UP 2001.

MIR 2.5.4, 2.8