This week’s topics

- Classical probabilistic retrieval model
  - Probability ranking principle, etc.
- 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.

Why probabilities in IR?

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

The document ranking problem

- 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

Why use probabilities?

- Information Retrieval deals with uncertain information
  - Infer whether a document is relevant to a user’s information need (imperfectly expressed by a query)
- Probability theory seems the most natural way to quantify uncertainty
- The classical probabilistic IR approach attempts to model precisely this:
  - \( P(\text{relevant} | \text{document}, \text{query}) \)

Why use probabilities?

Standard IR techniques

- Empirical for most part
  - Success measured by experimental results
- Few properties provable
  - Sometimes you want to analyze properties of methods

Probabilistic IR

- Probability Ranking Principle
  - Provable “minimization of risk”
- Probabilistic Inference
  - "Justify" your decision
- Nice theory
  - Performance benefits unclear
Recall a few probability basics

- Bayes' Rule
  \[ p(a,b) = p(a \cap b) = p(a \mid b) p(b) = p(b \mid a) p(a) \]
  \[ p(\overline{a} \mid b) p(b) = p(b \mid \overline{a}) p(\overline{a}) \]
  \[ p(a \mid b) = \frac{p(b \mid a) p(a)}{p(b)} = \sum_{x \in \mathbb{X}} p(b \mid x) p(x) \]
  Posterior
  - Odds:
    \[ O(y) = \frac{p(y)}{p(\overline{y})} = \frac{p(y)}{1 - p(y)} \]

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."


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.

\[ p(R \mid x) \] - probability that a document \( x \) is relevant.

Need to find \( p(R \mid x) \) - prior probability of retrieving a (non) relevant document

\[ p(R \mid x) = \frac{p(x \mid R) p(R)}{p(x)} = \frac{p(R, p(NR) \mid p(R) \mid p(R) + p(NR) \mid p(R)} \]

\[ p(R \mid x) + p(NR \mid x) = 1 \]

\[ p(R, p(NR) \mid p(R) \mid p(R) \]

\[ p(R \mid x) \quad \text{if a relevant (non-relevant) document is retrieved, it is } x. \]

Probability Ranking Principle

- Simple case: no selection costs or other utility concerns
- Bayes' Decision Rule
  - \( x \) is relevant \iff \( p(R \mid x) > p(NR \mid x) \)
- PRP in action: Rank all documents by \( p(R \mid x) \)
- Theorem:
  - Using the PRP is optimal, in that it minimizes the loss (Bayes risk) under 1/0 loss
  - Provable if all probabilities correct, etc.

Probability Ranking Principle

- How do we compute all those probabilities?
- Do not know exact probabilities, have to use estimates
- Binary Independence Retrieval (BIR) is 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 tf, df, and length influence your judgments about document relevance? (Okapi)
- Combine to find document relevance probability
- Order documents by decreasing probability

Binary Independence Model

- Traditionally used in conjunction with PRP
- “Binary” = Boolean: documents are represented as binary vectors of terms:
  - \( \vec{x} = (x_1, \ldots, x_n) \)
  - \( x_i = 1 \) if 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)

Queries: binary vectors of terms

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

\[
O(R|q,X) = \frac{p(R|q,X)}{p(NR|q,X)} = \frac{p(R|q)}{p(NR|q)} \cdot \frac{p(R|\vec{x},R,q)}{p(R|\vec{x},NR,q)}
\]

• Using Independence Assumption:
  - \( p(\vec{x}|R,q) = \prod_{i=1}^{n} p(x_i|R,q) \)
  - \( p(\vec{x}|NR,q) = \prod_{i=1}^{n} p(x_i|NR,q) \)
  - So: \( O(R|q,d) = O(R|q) \cdot \prod_{i=1}^{n} \frac{p(x_i|R,q)}{p(x_i|NR,q)} \)

Then...

\[ p_{1} = p_{1} \cdot \left( 1 - r_{1} \right) \quad \text{All matching terms} \]

\[ p_{2} = p_{2} \cdot \left( 1 - r_{2} \right) \quad \text{All query terms} \]

• Assume, for all terms not occurring in the query \( \vec{x} = 0 \): \( p_{2} = r_{2} \)
Binary Independence Model

\[ O(R|q, x) = O(R|q) \prod_{i=1}^{m} p_i(1-r_i) \prod_{i=1}^{m} (1-p_i) \]

Constant for each query

Only quantity to be estimated for rankings

• Retrieval Status Value:

\[ RSV = \log \prod_{k=q+1}^{m} \frac{p_i(1-r_i)}{r_i(1-p_i)} = \sum \log \frac{p_i(1-r_i)}{r_i(1-p_i)} \]

Estimation – key challenge

• If non-relevant documents are approximated by whole collection, then \( r_i = n/N \)
  \[ \log (1-r_i)/r_i = \log (N-n)/n = \log N/n = \log N - \log n = \log N/n \]

• \( p_i \) (probability of occurrence in relevant documents) can be estimated in various ways:
  - from relevant documents if know some
  - Relevance weighing can be used in feedback loop
  - constant (Croft and Harper combination match) – then just get idf weighting of terms
  - proportional to probability of occurrence in collection
  - more accurately, to log of this (Greiff, SIGIR 1998)

Removing term independence

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

PRP and BIR

• Getting reasonable approximations of probabilities is possible.
• Requires restrictive assumptions:
  - \textit{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 interior term weights
Bayesian Networks for Text Retrieval

- 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

Toy Example

- \( f \) 0.3 
- \( \neg f \) 0.7

- \( n \) 0.9 0.3 
- \( \neg n \) 0.1 0.7

- \( g \) 0.9 0.3 
- \( \neg g \) 0.1 0.7

- \( t \) 0.99 0.1 
- \( \neg t \) 0.01 0.9

Project Due (d)
- 0.4 
- \( \neg d \) 0.6

Finals (f)
No Sleep (n)
Gloom (g)
Triple Latte (t)

Links as dependencies

- Conditional Probability Table = Link Matrix
  - Attached to each node
    - Give influences of parents on that node.
  - Nodes with no parent get a “prior probability”
    - e.g., \( f, d \)
  - interior node: conditional probability of all combinations of values of its parents
    - e.g., \( n, g, t \)

Independence Assumption

- Variables not connected by a link: no direct conditioning.
- Joint probability - obtained from link matrices.
- See examples on next slide.

Independence Assumption

- Independence assumption:
  \( P(g|f) = P(g) \)
- Joint probability
  \( P(d|n,g,t) = P(d)P(n)P(g)P(t) \)
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
  - Polytree networks are easily tractable
  - Much other work on efficient exact and approximate Bayesian network inference

Diagnostic Inference
- Propagate beliefs through parents of a node
- Inference rule

\[
P(a | c) = \frac{\sum \ P(c | b_i)P(b_i | a)}{P(c)}
\]

Diagnostic inference
- Propagate beliefs through parents of a node
- Inference rule

```
P(f | n) = 0.25
P(\neg f | n) = 0.21
```

Diagnostic inference
- Propagate beliefs through parents of a node
- Inference rule

```
P(f | n) = 0.56
P(\neg f | n) = 0.44
```

Diagnostic inference
- Propagate beliefs through parents of a node
- Inference rule

```
P(f | n) = 0.56
P(\neg f | n) = 0.44
```

Diagnostic inference
- Propagate beliefs through parents of a node
- Inference rule

```
P(f | n) = 0.56
P(\neg f | n) = 0.44
```

Diagnostic inference
- Propagate beliefs through parents of a node
- Inference rule

```
P(f | n) = 0.56
P(\neg f | n) = 0.44
```

Diagnostic inference
- Propagate beliefs through parents of a node
- Inference rule

```
P(f | n) = 0.56
P(\neg f | n) = 0.44
```

Predictive Inference
- Compute belief of child nodes of evidence
- Inference rule

\[
P(c | a) = \sum \ P(c | b_i)P(b_i | a)
\]

Predictive Inference
- Compute belief of child nodes of evidence
- Inference rule

```
P(f | n) = 0.56
P(\neg f | n) = 0.44
```

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

Large, but compute once for each document collection.

- Document Network
  - \( d \)-documents
  - \( t_i \)-document representations
  - \( ri \)-“concepts”

- Query Network
  - \( c_1 \)-query concepts
  - \( q_i \)-high-level concepts
  - \( r_1 \)-goal node

Small, compute once for every query.

Bayesian Nets for IR: Roadmap

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

Bayesian nets for text retrieval

- \( d_i \): Documents
- \( t_i \): Terms
- \( c_i \): Concepts
- \( q_i \): Query operators (AND/OR/NOT)
- \( r \): Information need

- Prior probability \( P(d) \)
  - \( = 1/n \)
- \( P(r|d) \)
  - within-document term frequency
  - \( tf \times idf \)-based

Link matrices and probabilities

- \( P(q|r) \)
  - 1-to-1
  - thesaurus
- \( P(q|c) \): canonical forms of query operators
  - Always use things like AND and NOT – never store a full CPT*.

Example

- \( d_i \): Documents
- \( t_i \): Terms
- \( c_i \): Concepts
- \( q_i \): Query operators (AND/OR/NOT)
- \( r \): Information need

- 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”

Extensions

*conditional probability table
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

Exercise

- Consider ranking docs for a 1-term query. What is the difference between
  - A cosine-based vector-space ranking where each doc has $tf \times idf$ components, normalized;
  - A Bayes net in which the link matrices on the docs-to-term links are normalized $tf \times idf$?

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

  - http://www.dcs.gla.ac.uk/Kay/IR/relevance.html
  - http://www.is.t.u-tokyo.ac.jp/Research/personal/people/1998-30-4/content-
  - http://www.research.microsoft.com/~heckerman/
- MIR 2.5.4, 2.8

Resources

- E. Charniak. Bayesian nets without tears. AI Magazine 12(4): 50-63
  http://www.research.microsoft.com/~heckerman/
- MIR 2.5.4, 2.8