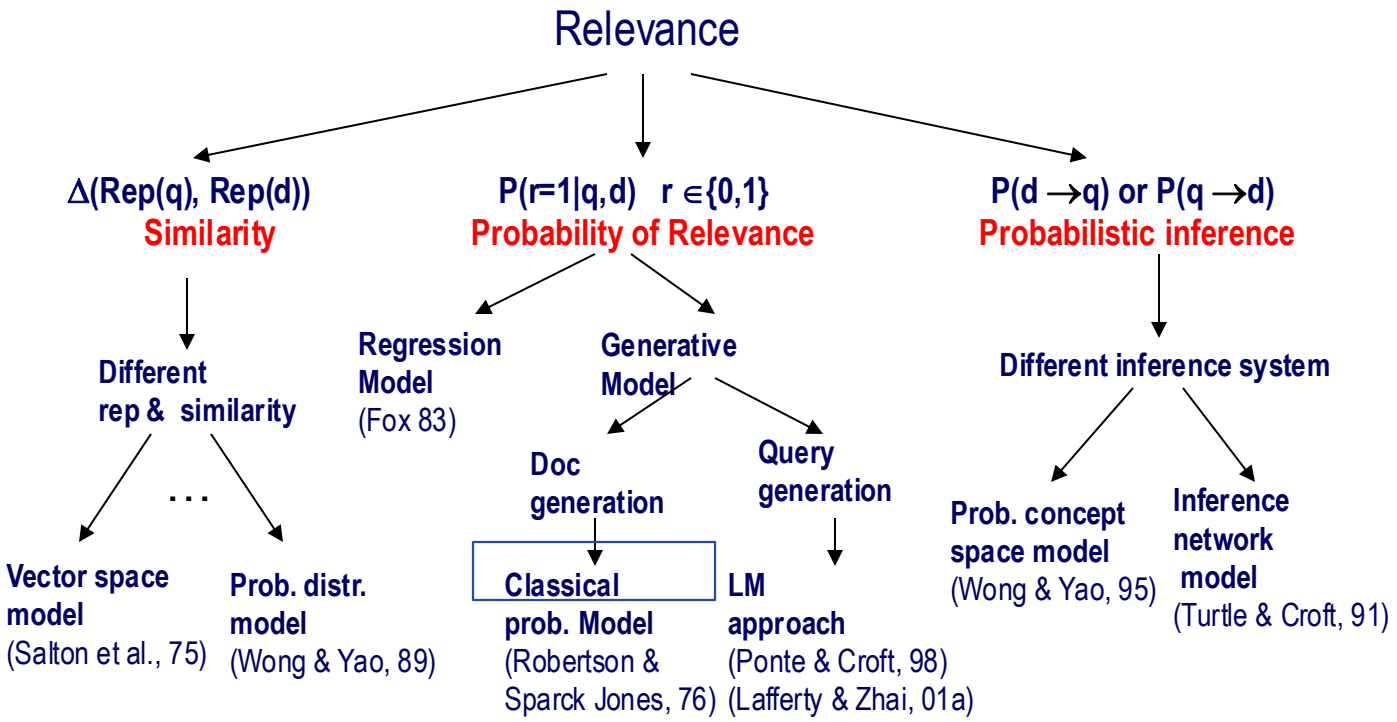
Information Retrieval

Probabilistic Information Retrieval

Notion of relevance



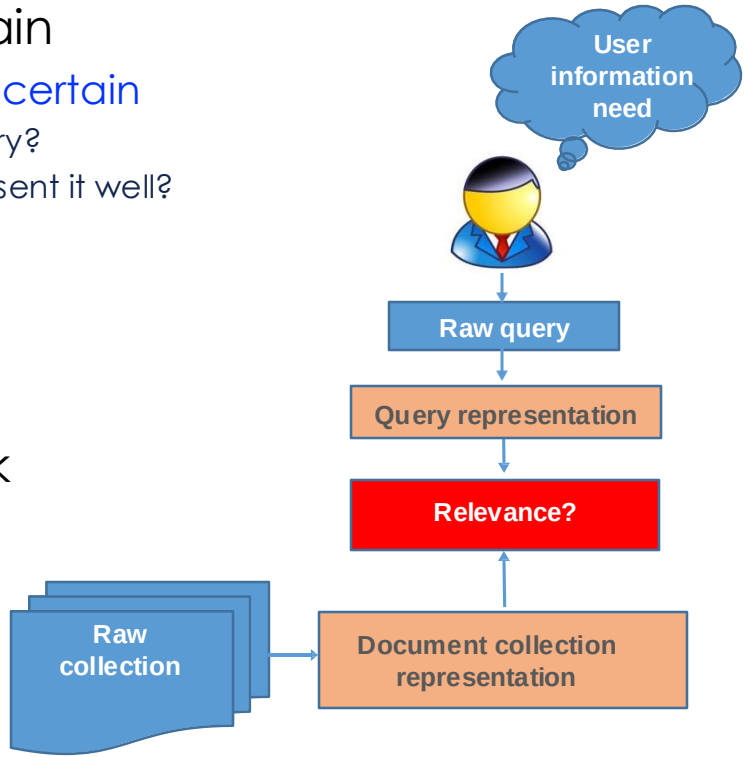
Outline

- Probabilistic Ranking Principle (PRP)
- Basics of probability theory
- Probabilistic ranking (log-odds)
- Binary Independence Model (BIM)
- BestMatch25 (BM25, Okapi)

Why probability theory in IR?

- As a process, retrieval is inherently uncertain
  - Understanding of user's information needs is uncertain
    - Are we sure the user mapped his need into a good query?
    - Even if the query represents well the need, did we represent it well?
  - Estimating document relevance for the query
    - Uncertainty from selection of document representation
    - Uncertainty from matching query and documents

- Probability theory is a common framework for modeling uncertainty
  - Idea: Rank by the probability of relevance of the document w.r.t. information need
    - P(R=1 | document, query)



Basic Idea

Query	Doc	Relevance
q1	d1	1
q1	d2	0
q1	d3	1
q1	d1	0
q1	d2	0
q2	d1	1
q2	d2	0
q3	d1	1
q4	d2	0

Basic Idea

Query	Doc	Relevance
q1	d1	1
q1	d2	0
q1	d3	1
q1	d1	0
q1	d2	0
q2	d1	1
q2	d2	0
q3	d1	1
q4	d2	0

$$\text{Score}(d,q) = p(R=1|d,q)$$
$$= \text{count}(q,d,R=1) / \text{count}(q,d)$$

$$P(R=1|q1,d1) = 1/2$$

$$P(R=1|q1,d2) = 0/2$$

$$P(R=1|q1,d3) = 1/1$$

Probabilistic vs. Other Models

- Boolean model:
  - Probabilistic models support ranking and thus are better than the simple Boolean model
- Vector space model:
  - The vector space model is also a formally defined model that supports ranking
  - Ranks documents according to similarity
    - The notion of similarity does not translate directly into an assessment of "is the document a good document to give to the user or not?"
    - The most similar document can be highly relevant or completely non-relevant
- Probability theory is arguably a cleaner formalization of what we want an IR system to do

Probabilistic approach to retrieval

- An IR system is primarily uncertain about
  - Understanding of the query
  - Whether a document satisfies the query
- Probability theory
  - Provides the foundation for reasoning under uncertainty
  - Probabilistic information retrieval models estimate how likely it is that a document is relevant for a query
- Probabilistic IR models
  - Classic probabilistic models (BIM, Two Poisson, BM11, BM25)
  - Language modelling for IR (next lectures)
- Probabilistic IR models are among the best-performing and most widely used IR models

Probabilistic Ranking Principle

- "If the retrieved documents are ranked decreasingly on their probability of relevance, then the effectiveness of the system will be the best that is obtainable"
- "If [the IR] system's response to each [query] is a ranking of the documents in order of decreasing probability of relevance to the [query], where the probabilities are estimated as accurately as possible based on 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 based on those data"
- "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."


Stephen Robertson


Karen Spärck Jones


Van Rijsbergen

Probabilistic Ranking Principle

- Assume the ranked retrieval setting
  - We are given a query *q* and a document collection *D*
  - Ordered list of documents from *D* is to be returned for *q*
- We model relevance (non-relevance) as **random binary variables**
  - $R_{d,q} = 1$ if document *d* from *D* is relevant for query *q*,
  - $R_{d,q} = 0$ otherwise
- **Probabilistic ranking principle**: The information retrieval system will reach the best obtainable efficiency if the documents are ranked decreasingly according to their probability of relevance
  - I.e., decreasingly in terms of $P(R_{d,q} = 1)$, or, equivalently, $P(R = 1 \mid d, q)$

Basic Probability Theory

For events A and B:

- Joint probability $P(A \cap B)$: both events occurring
- Conditional probability $P(A|B)$: A occurring given B has occurred

Chain rule gives relationship between joint/conditional probabilities:

$$P(AB) = P(A \cap B) = P(A|B)P(B) = P(B|A)P(A)$$

Similarly for the complement of an event $P(\overline{A})$:

$$P(\overline{A}B) = P(B|\overline{A})P(\overline{A})$$

Partition rule: if B can be divided into an exhaustive set of disjoint subcases, then $P(B)$ is the sum of the probabilities of the subcases. A special case of this rule gives:

$$P(B) = P(AB) + P(\overline{A}B)$$

Basic Probability Theory

Bayes' Rule for inverting conditional probabilities:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} = \frac{P(B|A)}{\sum_{X\in\{A,\overline{A}\}} P(B|X)P(X)} \cdot P(A)$$

Can be thought of as a way of updating probabilities:

- Start off with prior probability $P(A)$ (initial estimate of how likely event A is in the absence of any other information).
- Derive a posterior probability $P(A|B)$ after having seen the evidence B, based on the likelihood of B occurring in the two cases that A does or does not hold.

Odds of an event is a kind of multiplier for how probabilities change:

$$\text{Odds:} \qquad O(A) = \frac{P(A)}{P(\overline{A})} = \frac{P(A)}{1 - P(A)}$$

Often, instead of raw odds, we compute the logarithm of the odds, log-odds (for numeric convenience)

log(O(A)) = log P(A) – log(1-P(A))

Binary Independence Model

- Traditionally used in conjunction with PRP with the following assumptions:
  - "Binary" (equivalent to Boolean): documents are represented as binary incidence vectors of terms
    - e.g., document *d* represented by vector $x = (x_1, \ldots, x_M)$, where $x_t = 1$ if term t occurs in d and $x_t = 0$ otherwise
    - Different documents can be modeled as the same vector
  - "Independence": the presence or absence of a word in a document is independent of the presence or absence of any other word
    - not true, but works in practice

Binary Independence Matrix

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
ANTHONY	1	1	0	0	0	1	
BRUTUS	1	1	0	1	0	0	
CAESAR	1	1	0	1	1	1	
CALPURNIA	0	1	0	0	0	0	
CLEOPATRA	1	0	0	0	0	0	
MERCY	1	0	1	1	1	1	
WORSER	1	0	1	1	1	0	
...							

Each document is represented as a binary vector $\in \{0,1\}^{|V|}$.

Binary Independence Model

- To make a probabilistic retrieval strategy precise, we need to estimate how terms in documents contribute to relevance:
  - Find measurable statistics (term frequency, document frequency, document length) that affect judgments about document relevance.
  - Combine these statistics to estimate the probability $P(R|d, q)$ of document relevance.

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:
  - Odds = the probability that the event will occur divided by the probability that the event will not occur, i.e., *p/(1-p)*.

$$O(R \mid q, \vec{x}) = \frac{p(R=1 \mid q, \vec{x})}{p(R=0 \mid q, \vec{x})} = \frac{\dfrac{p(R=1 \mid q)\, p(\vec{x} \mid R=1, q)}{p(\vec{x} \mid q)}}{\dfrac{p(R=0 \mid q)\, p(\vec{x} \mid R=0, q)}{p(\vec{x} \mid q)}}$$

Binary Independence Model

$$O(R \mid q, \vec{x}) = \frac{p(R=1 \mid q, \vec{x})}{p(R=0 \mid q, \vec{x})} = \frac{p(R=1 \mid q)}{p(R=0 \mid q)} \times \frac{p(\vec{x} \mid R=1, q)}{p(\vec{x} \mid R=0, q)}$$

Constant for a given query, can be ignored

Needs estimation

– Using Independence Assumption:

$$O(R \mid q, \vec{x}) = O(R \mid q) \times \prod_{i=1}^{n} \frac{p(x_i \mid R=1, q)}{p(x_i \mid R=0, q)}$$

– Since x_i is either 0 or 1, we can separate the terms:

$$O(R \mid q, \vec{x}) = O(R \mid q) \times \prod_{x_i=1} \frac{p(x_i=1 \mid R=1, q)}{p(x_i=1 \mid R=0, q)} \times \prod_{x_i=0} \frac{p(x_i=0 \mid R=1, q)}{p(x_i=0 \mid R=0, q)}$$

Binary Independence Model

- Let be the probability of a term appearing in a **relevant** document

$$p_i = p(x_i = 1 | R = 1, q);$$

- And the probability of a term appearing in a **nonrelevant** document.

$$r_i = p(x_i = 1 | R = 0, q);$$

- Assume, that all terms not occurring in the query they are equally likely to occur in relevant and nonrelevant documents

$$p_i = r_i$$

- We only consider terms in the products that appear in the query:

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

Terms in document

Terms not in document

Binary Independence Model

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

$$O(R \mid q, x) = O(R \mid q) \times \prod_{\substack{x_i = 1 \\ q_i = 1}} \frac{p_i}{r_i} \times \prod_{\substack{x_i = 1 \\ q_i = 1}} \left(\frac{1 - r_i}{1 - p_i} \times \frac{1 - p_i}{1 - r_i} \right) \times \prod_{\substack{x_i = 0 \\ q_i = 1}} \frac{1 - p_i}{1 - r_i}$$

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

$x_i = q_i = 1$: All matching terms

$q_i = 1$: All query terms

Binary Independence Model

Constant for each query

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

The only quantity to be estimated for rankings

- We can equally rank documents by the logarithm since log is a monotonic function.
  - Hence the Retrieval Status Value (RSV) in this model:

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

Binary Independence Model

- Equivalent: rank documents using log odds ratios for query terms c_i:

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

$p_i/(1- p_i)$ odds of the term appearing if the document is relevant

$r_i/(1- r_i)$ odds of the term appearing if the document is not relevant

  - *or* $\log(p_i/(1-p_i)) - \log(r_i/(1-r_i))$
  - $c_i = 0$: term has equal odds of appearing in relevant and nonrel. docs.
  - c_i positive: higher odds to appear in relevant documents.
  - c_i negative: higher odds to appear in nonrelevant documents.
- So BIM and vector space model are identical on an operational level
  - … except that the term weights are different.
- In particular: we can use the same data structures (inverted index etc)for the two models.

Key challenge: estimation

$$c_i = \log \frac{p_i(1-r_i)}{r_i(1-p_i)}$$

- 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 (i.e., df_i/N) and

$$\log \frac{1 - r_i}{r_i} = \log \frac{N - n - S + s}{n - s} \approx \log \frac{N - n}{n} \approx \log \frac{N}{n} = IDF!$$

Key challenge: estimation

- p_i (probability of occurrence in relevant documents) cannot be approximated as easily
- p_i can be estimated in various ways:
  - from relevant documents if you know some
    - Relevance weighting can be used in a feedback loop
  - constant (Croft and Harper combination match model) –with p_i=0.5, this is simply idf weighting of query terms

$$RSV = \sum_{x_i=q_i=1} \log \frac{N}{n_i}$$

  - proportional to prob. of occurrence in collection
    - E.g., 1/3 + 2/3 df_i/N

Probabilistic Relevance Feedback

- Guess a preliminary probabilistic model of R=1 documents; use it to retrieve a set of documents
- Interact with the user to refine the model: partition subset V into VR and VNR
- Re-estimate p_i and r_i on the basis of these
  - If i appears in VRi within set of documents VR: pi = | VRi | / | VR |
  - Or combine new information with previous guess (Bayesian update):

$$p_i^{(n+1)} = \frac{|V_{Ri}| + \kappa p_i^{n}}{|V_R| + \kappa}$$

p_i – *Bayesian prior*
κ - prior weight

- Repeat, thus generating a succession of approximations to relevant documents

Iteratively estimating p_i and r_i (= Pseudo-relevance feedback)

1. Assume that p_i is constant over all x_i in query and r_i as before
   - 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
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

PRP and BIM

- 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

- Problem: either requires partial relevance information or, seemingly, it can only derive somewhat inferior term weights

A key limitation of the BIM

- BIM – like much of original IR – was designed for titles or abstracts, and not for modern full text search
- We want to pay attention to term frequency and document lengths, just like in other models we discuss
- Want some model of how often terms occur in docs

$$c_i = \log \frac{p_{tf} r_0}{p_0 r_{tf}}$$

How different are vector space model and BIM?

- They are not that different.
- In either case, we build an information retrieval scheme in the exact same way.
- For probabilistic IR, at the end, we score queries not by cosine similarity and tf-idf in a vector space, but by a slightly different formula motivated by probability theory.

- Next: how to add term frequency and length normalization to the probabilistic model.

BM25

- Okapi BM25 is a probabilistic model that incorporates term frequency (i.e., it's nonbinary) and length normalization.
- BIM was originally designed for short catalog records of fairly consistent length, and it works reasonably in these contexts.
- For modern full-text search collections, a model should pay attention to term frequency and document length.

- BestMatch25 (a.k.a BM25 or Okapi).
  - Goal: be sensitive to term frequency and document length while not adding too many parameters
  - 25 because they had a bunch of tries!
- BM25 is one of the most widely used and robust retrieval models.

Document length normalization

- Longer documents are likely to have larger tf_i values
- Why might documents be longer?
  - Verbosity: suggests observed tf_i too high
  - Larger scope: suggests observed tf_i may be right
- A real document collection probably has both effects
- … so should apply some kind of partial normalization

Document length normalization

- Document length: $dl = \sum_{i \in V} tf_i$
- *avdl*: Average document length over collection
- Length normalization component

$$B = \left((1 - b) + b \frac{dl}{avdl}\right) \qquad 0 \leq b \leq 1$$

  - b = 1 full document length normalization
  - b = 0 no document length normalization

BM25

$$RSV^{BM25} = \sum_{i \in q} \log \frac{N}{df_i} \times \frac{(k_1+1)tf_i}{k_1((1-b)+b\frac{dl}{avdl})+tf_i}$$

- k_1 controls term frequency scaling
  - k_1 = 0 is binary model; k_1 large is raw term frequency
- b controls document length normalization
  - b = 0 is no length normalization;
  - b = 1 is relative frequency (fully scale by document length)

- Typically, k_1 is set around 1.2–2 and b around 0.75

Summary

- Differences between probabilistic IR models and vector space model
  - Different theoretical underpinnings, but similar ranking effects
  - The ranking function of the probabilistic models is grounded in probability theory
  - The ranking function of VSM – cosine similarity – is grounded in vector algebra

- Binary independence model – binary term weights
  - Similar effect to ignoring the TF component in VSM
    - i.e., just IDF weighting
- BM25 – dampens the effects of document length
  - Similar to taking a logarithm of length-normalized frequency as TF in VSM
    - $\log(f_{t},D \,/\, \max f_{t',}D)$