Introduction to Information Retrieval

Probabilistic Information Retrieval

Overview

- Probabilistic Approach to Retrieval
- Basic Probability Theory
- Probability Ranking Principle
- Appraisal & Extensions

Probabilistic Approach To Retrieval

Probabilistic Approach to Retrieval

- Given a user information need (query) and a collection of documents (transformed into document representations), a system must determine how well the documents satisfy the query
- An IR system has an uncertain understanding of the user query , and makes an uncertain guess of whether a document satisfies the query
- Probability theory provides a principled foundation for such reasoning under uncertainty
- Probabilistic models exploit this foundation to estimate how likely it is that a document is relevant to a query

- Classical probabilistic retrieval model -- Probability ranking principle

Basic Probability Theory

Basic Probability Theory

- For events A and B
  - P(*A*, *B*): Joint probability of both events occurring
  - P(*A*|*B*) Conditional probability of event *A* occurring given event B has occurred
- Chain rule gives fundamental relationship between joint and conditional probabilities:

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

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

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

Basic Probability Theory

- **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:

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

Basic Probability Theory

- Bayes' Rule for inverting conditional probabilities:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} = \left[\frac{P(B|A)}{\sum_{X\in\{A,\overline{A}\}} P(B|X)P(X)}\right] P(A)$$

- It 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

Basic Probability Theory

- Odds of an event provide a kind of multiplier for how probabilities change:
    - Odds: $$O(A) = \frac{P(A)}{P(\overline{A})} = \frac{P(A)}{1 - P(A)}$$

Probability Ranking Principle

The Document Ranking Problem

- **Ranked retrieval setup**: given a collection of documents,
  - user issues a query
  - an ordered list of documents is returned
- **Assume binary notion of relevance**: $R_{d,q}$ is a random dichotomous variable, such that
  - $R_{d,q}$ = 1, if document d is relevant w.r.t query q
  - $R_{d,q}$ = 0, otherwise
- Probabilistic ranking orders documents decreasingly by their estimated probability of relevance w.r.t. query: **P(R = 1|d, q)**

Probability Ranking Principle (PRP)

- **PRP in brief**
  - If the retrieved documents (w.r.t a query) are ranked decreasingly on their probability of relevance, then the effectiveness of the system will be the best that is obtainable
- **PRP in full**
  - 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 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 (PRP)

- Optima Decision Rule

$$d \text{ is relevant iff } P(R=1|d,q) > P(R=0|d,q)$$

- Let C_1 be the cost of retrieving a relevant document and C_0 be the cost of retrieval of a non-relevant document. Then Probability Ranking Principle says that if for a specific document d, and for all documents d', not yet retrieved

$$C_0 \cdot P(R=0|d) + C_1 \cdot P(R=1|d) \leq C_0 \cdot P(R=0|d') + C_1 \cdot P(R=1|d')$$

- For C_1 < C_0, cost is minimized by choosing argmax_d [P(R=1|d)]

Binary Independence Model (BIM)

- Traditionally used with the PRP
- Assumptions:
  - ‘**Binary**’ (equivalent to Boolean): documents and queries represented as binary term incidence vectors
    - E.g., document *d* represented by vector $\vec{x} = (x_1, \ldots, x_M)$, where $x_t = 1$ if term *t* occurs in *d* and $x_t = 0$ otherwise
    - Different documents may have the same vector representation
  - ‘**Independence**’: no association between terms (not true, but practically works - ‘naive’ assumption of Naive Bayes models)

Binary Independence Model (BIM)

- To make a probabilistic retrieval strategy precise, 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

Binary Independence Model (BIM)

- How terms in documents contribute to relevance (continued)
  - Combine these statistics to estimate the probability of document relevance
  - Order documents by decreasing estimated probability of relevance $P(R|d, q)$
  - Assume that the relevance of each document is independent of the relevance of other documents (not true, in practice allows duplicate results)

Binary Independence Model (BIM)

$P(R|d,q)$ is modelled using term incidence vectors as $P(R|\vec{x},\vec{q})$

$$P(R=1|\vec{x},\vec{q}) = \frac{P(\vec{x}|R=1,\vec{q})P(R=1|\vec{q})}{P(\vec{x}|\vec{q})}$$

$$P(R=0|\vec{x},\vec{q}) = \frac{P(\vec{x}|R=0,\vec{q})P(R=0|\vec{q})}{P(\vec{x}|\vec{q})}$$

$P(\vec{x}|R=1,\vec{q})$ and $P(\vec{x}|R=0,\vec{q})$: probability that if a relevant or non-relevant document is retrieved, then that document's representation $\vec{x}$; Statistics about the actual document collection are used to estimate these probabilities

Binary Independence Model (BIM)

$P(R|d,q)$ is modelled using term incidence vectors as $P(R|\vec{x},\vec{q})$

$$P(R=1|\vec{x},\vec{q}) = \frac{P(\vec{x}|R=1,\vec{q})P(R=1|\vec{q})}{P(\vec{x}|\vec{q})}$$

$$P(R=0|\vec{x},\vec{q}) = \frac{P(\vec{x}|R=0,\vec{q})P(R=0|\vec{q})}{P(\vec{x}|\vec{q})}$$

$P(R=1|\vec{q})$ and $P(R=0|\vec{q})$ prior probability of retrieving a relevant or non-relevant document for a query *q*

Estimate $P(R=1|\vec{q})$ and $P(R=0|\vec{q})$ from percentage of relevant documents in the collection

Since a document is either relevant or non-relevant to a query, we must have that: $P(R=1|\vec{x},\vec{q}) + P(R=0|\vec{x},\vec{q}) = 1$

Deriving a Ranking Function for Query Terms

Given a query q, ranking documents by $P(R=1|d,q)$ is modeled under BIM as ranking them by $P(R=1|\vec{x},\vec{q})$

Easier: rank documents by their odds of relevance (gives same ranking & we can ignore the common denominator)

$$O(R|\vec{x},\vec{q}) = \frac{P(R=1|\vec{x},\vec{q})}{P(R=0|\vec{x},\vec{q})} = \frac{\frac{P(R=1|\vec{q})P(\vec{x}|R=1,\vec{q})}{P(\vec{x}|\vec{q})}}{\frac{P(R=0|\vec{q})P(\vec{x}|R=0,\vec{q})}{P(\vec{x}|\vec{q})}}$$

$$= \frac{P(R=1|\vec{q})}{P(R=0|\vec{q})} \cdot \frac{P(\vec{x}|R=1,\vec{q})}{P(\vec{x}|R=0,\vec{q})}$$

$\frac{P(R=1|\vec{q})}{P(R=0|\vec{q})}$ is a constant for a given query - can be ignored

Deriving a Ranking Function for Query Terms

- It is at this point that we make the Naive Bayes conditional independence assumption that the presence or absence of a word in a document is independent of the presence or absence of any other word (given the query):

$$\frac{P(\vec{x}|R=1,\vec{q})}{P(\vec{x}|R=0,\vec{q})} = \prod_{t=1}^{M} \frac{P(x_t|R=1,\vec{q})}{P(x_t|R=0,\vec{q})}$$

- So:

$$O(R|\vec{x},\vec{q}) = O(R|\vec{q}) \cdot \prod_{t=1}^{M} \frac{P(x_t|R=1,\vec{q})}{P(x_t|R=0,\vec{q})}$$

Deriving a Ranking Function for Query Terms

- Since each x_t is either 0 or 1, we can separate the terms to give:

$$O(R|\vec{x},\vec{q}) = O(R|\vec{q}) \cdot \prod_{t:x_t=1} \frac{P(x_t=1|R=1,\vec{q})}{P(x_t=1|R=0,\vec{q})} \cdot \prod_{t:x_t=0} \frac{P(x_t=0|R=1,\vec{q})}{P(x_t=0|R=0,\vec{q})}$$

  - Let $p_t = P(x_t = 1|R = 1, \vec{q})$ be the probability of a term appearing in relevant document
  - Let $u_t = P(x_t = 1|R = 0, \vec{q})$ be the probability of a term appearing in a non-relevant document

- Visualise as contingency table:

	document	relevant ($R=1$)	nonrelevant ($R=0$)
Term present	$x_t = 1$	p_t	u_t
Term absent	$x_t = 0$	$1 - p_t$	$1 - u_t$

Deriving a Ranking Function for Query Terms

- **Additional simplifying assumption**: terms not occurring in the query are equally likely to occur in relevant and non-relevant documents
  - If $q_t = 0$, then $p_t = u_t$
- Now we need only to consider terms in the products that appear in the query:

$$P(q|M_d) = P(\langle t_1, \ldots, t_{|q|} \rangle | M_d) = \prod_{1 \leq k \leq |q|} P(t_k|M_d)$$

  - The left product is over query terms found in the document and the right product is over query terms not found in the document

Deriving a Ranking Function for Query Terms

Let us make an additional simplifying assumption that terms not occurring in the query are equally likely to occur in relevant and nonrelevant documents: that is, if $q_t = 0$ then $p_t = u_t$. (This assumption can be changed, as when doing relevance feedback in Section 11.3.4.) Then we need only consider terms in the products that appear in the query, and so,

$$O(R|\vec{q},\vec{x}) = O(R|\vec{q}) \cdot \prod_{t:x_t=q_t=1} \frac{p_t}{u_t} \cdot \prod_{t:x_t=0,q_t=1} \frac{1-p_t}{1-u_t}$$

Deriving a Ranking Function for Query Terms

- Including the query terms found in the document into the right product, but simultaneously dividing through by them in the left product, gives:

$$O(R|\vec{x},\vec{q}) = O(R|\vec{q}) \cdot \prod_{t:x_t=q_t=1} \frac{p_t(1-u_t)}{u_t(1-p_t)} \cdot \prod_{t:q_t=1} \frac{1-p_t}{1-u_t}$$

    - The left product is still over query terms found in the document, but the right product is now over all query terms, hence constant for a particular query and can be ignored. The only quantity that needs to be estimated to rank documents w.r.t a query is the left product

Deriving a Ranking Function for Query Terms

- Hence the Retrieval Status Value (RSV) in this model:

$$RSV_d = \log \prod_{t:x_t=q_t=1} \frac{p_t(1-u_t)}{u_t(1-p_t)} = \sum_{t:x_t=q_t=1} \log \frac{p_t(1-u_t)}{u_t(1-p_t)}$$

Deriving a Ranking Function for Query Terms

- Computing the *RSV*:
  - We can equally rank documents using the log odds ratios for the terms in the query c_t :

$$c_t = \log \frac{p_t(1-u_t)}{u_t(1-p_t)} = \log \frac{p_t}{(1-p_t)} + \log \frac{1-u_t}{u_t}$$

  - The odds ratio is the ratio of two odds:
    - The odds of the term appearing if the document is relevant ($p_t/(1 - p_t)$)
    - The odds of the term appearing if the document is non-relevant ($u_t/(1 - u_t)$)

Deriving a Ranking Function for Query Terms

- Computing the *RSV* (continued)

$$c_t = \log \frac{p_t(1-u_t)}{u_t(1-p_t)} = \log \frac{p_t}{(1-p_t)} + \log \frac{1-u_t}{u_t}$$

  - $c_t = 0$ if a term has equal odds of appearing in relevant and nonrelevant documents
  - $c_t > 0$ if it is more likely to appear in relevant documents
  - c_t functions as a term weight, so that $RSV_d = \sum_{x_t = q_t = 1} c_t$.
    Operationally, we sum c_t quantities in accumulators for query terms appearing in documents, just as for the vector space model calculations.

Deriving a Ranking Function for Query Terms

- For each term t in a query, estimate c_t in the whole collection using a contingency table of counts of documents in the collection.
- df_t is the number of documents that contain term t:

	documents	relevant	nonrelevant	Total
Term present	$x_t = 1$	s	$\text{df}_t - s$	df_t
Term absent	$x_t = 0$	$S - s$	$(N - \text{df}_t) - (S - s)$	$N - \text{df}_t$
	Total	S	$N - S$	N

$$p_t = s/S$$

$$u_t = (\text{df}_t - s)/(N - S)$$

$$c_t = K(N, \text{df}_t, S, s) = \log \frac{s/(S - s)}{(\text{df}_t - s)/((N - \text{df}_t) - (S - s))}$$

- To avoid the possibility of zeros (such as if every or no relevant document has a particular term) there are different ways to apply **smoothing.**

Exercise

- *Query: Obama health plan*
- *Doc1: Obama rejects allegations about his own bad health*
- *Doc2: The plan is to visit Obama*
- *Doc3: Obama raises concerns with US health plan reforms*

Estimate the probability that the above documents are relevant to the query.

Use a contingency table. These are the only three documents in the collection

Probability Estimates in Practice

- Assuming that relevant documents are a very small percentage of the collection, approximate **statistics for non-relevant documents** by **statistics from the whole collection**
- Let u_t is the probability of term occurrence in non-relevant documents for a query. Let us approximate,
  - $u_t \approx \mathrm{df}_t/N$
  - $\log[(1 - u_t)/u_t] = \log[(N - \mathrm{df}_t)/\mathrm{df}_t] \approx \log N/\mathrm{df}_t$
- Not easily extendable to relevant documents

$$c_t = \log \frac{p_t(1-u_t)}{u_t(1-p_t)} = \log \frac{p_t}{(1-p_t)} + \log \frac{1-u_t}{u_t}$$

$\log N/\mathrm{df}_t$

Probability Estimates in Practice

- Statistics of relevant documents (p_t) can be estimated in various ways:
  - Use the frequency of term occurrence in known relevant **documents (if known)**. This is the basis of probabilistic approaches to relevance feedback weighting in a feedback loop
  - **If not known**

Probability Estimates in Practice

- Statistics of relevant documents (p_t) estimation:
    - Set as constant. E.g., assume that pt is constant over all terms x_t in the query and that $p_t = 0.5$
        - Each term is equally likely to occur in a relevant document, and so the p_t and $(1 - p_t)$ factors cancel out in the expression for *RSV*
        - Weak estimate, but doesn't disagree violently with expectation that query terms appear in many but not all relevant documents
        - Combining this method with the earlier approximation for u_t, the document ranking is determined simply by which query terms occur in documents scaled by their idf weighting

$$c_t = \log \frac{p_t(1-u_t)}{u_t(1-p_t)} = \log \frac{p_t}{(1-p_t)} + \log \frac{1-u_t}{u_t} \quad \log N/\mathrm{df}_t$$

        - For short documents (titles or abstracts) in one-pass retrieval situations, this estimate can be quite satisfactory

Appraisal & Extensions

An Appraisal of Probabilistic Models

- Among the oldest formal models in IR
  - Maron & Kuhns, 1960: Since an IR system cannot predict with certainty which document is relevant, we should deal with probabilities
- Assumptions for getting reasonable approximations of the needed probabilities (in the BIM):
  - Boolean representation of documents/queries/relevance
  - Term independence
  - Out-of-query terms do not affect retrieval
  - Document relevance values are independent

An Appraisal of Probabilistic Models

- The difference between ‘vector space’ and ‘probabilistic’ IR is not that great:
  - In either case you build an information retrieval scheme in the exact same way.
  - Difference:
    - For probabilistic IR, at the end, you score queries not by cosine similarity and tf-idf in a vector space, but by a slightly different formula motivated by probability theory

Okapi BM25: A Nonbinary Model

- The 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) is sensitive to these quantities
- **BM25** has been **one of the most widely used and robust retrieval models**

Okapi BM25: A Nonbinary Model

- The simplest score for document d is just idf weighting of the query terms present in the document:

$$RSV_d = \sum_{t \in q} \log \frac{N}{\mathrm{df}_t}$$

- Improve this formula by factoring in the term frequency and document length:

$$RSV_d = \sum_{t \in q} \log \left[\frac{N}{\mathrm{df}_t} \right] \cdot \frac{(k_1 + 1)\mathrm{tf}_{td}}{k_1((1-b) + b \times (L_d / L_{\mathrm{ave}})) + \mathrm{tf}_{td}}$$

  - tf_{td} : term frequency in document d
  - L_d (L_{ave}): length of document d (average document length in collection)
  - k_1: tuning parameter controlling the document term frequency scaling
  - b: tuning parameter controlling the scaling by document length

Okapi BM25: A Nonbinary Model

- If the query is long, we might also use similar weighting for query terms

$$RSV_d = \sum_{t \in q} \left[\log \frac{N}{\mathrm{df}_t} \right] \cdot \frac{(k_1 + 1)\mathrm{tf}_{td}}{k_1((1-b) + b \times (L_d/L_{ave})) + \mathrm{tf}_{td}} \cdot \frac{(k_3 + 1)\mathrm{tf}_{tq}}{k_3 + \mathrm{tf}_{tq}}$$

  - tf_{tq}: term frequency in the query q
  - k_3: tuning parameter controlling term frequency scaling of the query
  - No length normalisation of queries (because retrieval is being done with respect to a single fixed query)
  - The above tuning parameters should ideally be set to optimize performance on a development test collection. In the absence of such optimisation, experiments have shown that k_1 and k_3 can be set to a value between 1.2 and 2 and b can be set to 0.75

Recap

- Probabilistically grounded approach to IR
- Probability Ranking Principle
- Models: BIM, BM25
- Assumptions

Thank you

Questions?