Probabilistic information retrieval

lightsilver

Feb 21st

Background

- Given only a query, an IR system has an uncertain understanding of the information need.
- Given the query and document representations, a system has an uncertain guess of whether a document has content relevant to the information need.
- Probability theory provides a principled foundation for such reasoning under uncertainty , exploiting which, estimating **how likely** it is that a **document** is relevant to an **information need**.

Retrieval models

- Probability Ranking Principle
- Binary Independence Model
  - most influential
- Okapi BM25 weighting scheme, and Bayesian Network models for IR

Review of probability theory

- Chain rule:
  - P(A, B) = P(A ∩ B) = P(A|B)P(B) = P(B|A)P(A)
- => $P(\overline{A}, B) = P(B|\overline{A})P(\overline{A})$
- Partition rule: $P(B) = P(A, B) + P(\overline{A}, B)$

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

- Bayes's rule: (a posterior probability)

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

Probability Ranking Principle

- The 1/0 loss case
  - For a query q and a document d in the collection, let $R_{d,q}$ be 1 if d is relevant with respect to q, 0 otherwise.
  - Estimated probability of relevance with respect to the information need: $P(R = 1|d, q)$.

Cases w/o retrieval costs

- Return best possible results as the top k documents
  - Simply rank all documents in decreasing order of $P(R = 1|d, q)$
- A set of retrieval results is to be returned, rather than an ordering:
  - Simply return documents that are more likely relevant than non-relevant
  - d is relevant iff $P(R = 1|d, q) > P(R = 0|d, q)$

Short point

- Is it possible for all probabilities are known correctly in practice?
  - Never.

Consider retrieval cost?

- Let C1 be the cost of retrieval of a relevant document and C0 the cost of retrieval of a non-relevant document.

- For a specific document d and for all documents d' not yet retrieved

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

Binary Independence Model

- Estimating the probability function P(R|d, q) is practical.
- Documents and queries are both represented as binary term incidence vectors.

$$\vec{x} = (x_1, \ldots, x_M)$$

- where $x_t = 1$ if term t is present in document d and $x_t = 0$ if t is not present in d.

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

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

- Statistics about the actual document collection are used to estimate these probabilities.
- $P(R=1|\vec{q})$ and $P(R=0|\vec{q})$ indicate the prior probability of retrieving a relevant or non-relevant document respectively for a query q.

Ranking function

- Rank documents by odds of relevance

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

- Naive Bayes conditional independence assumption :
  - The presence or absence of a word in a document is independent of the presence or absence of any other word

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

Ranking function cont.

- Expand:

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

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

- Retrieval Status Value (RSV)

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

- Ct: log odds ratios for the terms in the query.

How to estimate c_t quantities

- The c_t quantities function as term weights in the model, and the document score for a query is RSVd = $\sum_{x_t=q_t=1} c_t$.

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

$$\hat{c}_t = K(N, \mathrm{df}_t, S, s) = \log \frac{(s+\frac{1}{2})/(S-s+\frac{1}{2})}{(\mathrm{df}_t - s + \frac{1}{2})/(N - \mathrm{df}_t - S + s + \frac{1}{2})}$$

- simple form of smoothing

Bayesian prior

Probability estimates in practice

- Relevant documents are a very small percentage of the collection
- Approximate statistics for non-relevant documents by statistics from the whole collection.
- $u_t = df_t / N$

$$\log[(1-u_t)/u_t] = \log[(N-\mathrm{df}_t)/\mathrm{df}_t] \approx \log N/\mathrm{df}_t$$

Quantity p_t estimated

- Use the frequency of term occurrence in known relevant documents.
  - relevance feedback weighting in a feedback loop
- Using a constant in their combination match model.
  - pt = 0.5, 1-pt == pt, ut = dft / N
  - document ranking is determined simply by which query terms occur in documents scaled by their idf weighting.
- pt = dft/N.

Relevance feedback

- Guess initial estimates of pt and ut.
  - pt = 0.5?
- Use the current estimates of pt and ut to determine a best guess at the set of relevant documents $R = \{d : R_{d,q} = 1\}$.
  - Present to user & interact
- Refine the model of R
  - Learning from user relevance judgments for some subset of documents V.
  - VR = $\{d \in V, R_{d,q} = 1\} \subset R$
  - VNR = $\{d \in V, R_{d,q} = 0\}$,

Relevance feedback cont.I

- Re-estimate pt and ut on the basis of known relevant and non-relevant documents.

$$p_t = |VR_t|/|VR|$$

  - VRt is the set of documents in VR containing xt

$$p_t = \frac{|VR_t| + \frac{1}{2}}{|VR| + 1}$$

  - the set of documents judged by the user (V) is usually very small
  - resulting statistical estimate is quite unreliable (noisy)
  - Bayesian updating

$$p_t^{(k+1)} = \frac{|VR_t| + \kappa p_t^{(k)}}{|VR| + \kappa}$$

Relevance feedback cont.II

- Here $p_t^{(k)}$ is the k^{th} estimate for pt in an iterative updating process and is used as a Bayesian prior in the next iteration with a weighting of κ.
- κ = 5 is perhaps appropriate

- Repeat the above process from step 2, generating a succession of approximations to R and hence pt, until the user is satisfied.
- Pretend VR = V,

$$c_t = \log \frac{|V_t| + \frac{1}{2}}{|V| - |V_t| + 1} + \log \frac{N}{\mathrm{df}_t}$$

- Vector space
- Probabilistic

  - Either works