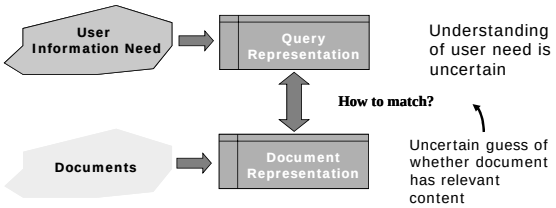
KOM341
Temu Kembali Informasi

KULIAH #11
- Probabilistic Information Retrieval

Probabilistic IR

- Pada boolean model atau vector space model, proses matching query dan dokumen dilakukan dengan menggunakan definisi formal melalui perhitungan indeks term.
- Masalah: sistem IR memiliki pengertian ketidakpastian dari informasi yang dibutuhkan. Artinya, sistem tidak tahu pasti apakah dokumen yang relevan dengan query memang sesuai dengan kebutuhan user.
- Dibutuhkan pendekatan peluang.

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?*

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 -- information need**
  - P(relevant|document$_i$, query)

Probabilistic relevance feedback

- Pada model IR sebelumnya, penekanan pada bobot term, sedangkan pada PIR menggunakan pendekatan peluang.
- Asumsikan kita sudah mengetahui ada beberapa dokumen relevan (R) dan beberapa dokumen tidak relevan (NR).
- P(t|R) adalah peluang munculnya term t pada dokumen yang relevan.
- P(t|NR) adalah peluang munculnya term t pada dokumen yang tidak relevan.

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

Probability Ranking Principle (PRP)

- Let *x* be a document in the collection.
- Let *R* represent relevance of a document given (fixed) query and let *NR* represent non-relevance.
- Need to find p(*R*|*x*) - probability that a document *x* is relevant.

$$p(R|x) = \frac{p(x|R)p(R)}{p(x)}$$

$$p(NR|x) = \frac{p(x|NR)p(NR)}{p(x)}$$

$$p(R|x) + p(NR|x) = 1$$

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 iff p(*R*|*x*) > p(*NR*|*x*)
- PRP in action: Rank all documents by p(*R*|*x*)

Probability Ranking Principle (PRP)

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

$$C \cdot p(R|d) + C' \cdot (1 - p(R|d)) \leq C \cdot p(R|d') + C' \cdot (1 - p(R|d'))$$

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

Probability Ranking Principle

- How do we compute all those probabilities?
  - Do not know exact probabilities, have to use estimates
  - Binary Independence Retrieval (BIR) – is the simplest model
- Questionable assumptions
  - "Relevance" of each document is independent of relevance of other documents.
  - Boolean model of relevance
  - That one has a single step information need

Binary Independence Model

- Traditionally used in conjunction with PRP
- **"Binary" = Boolean**: documents are represented as binary incidence vectors of terms:
  - $\vec{x} = (x_1, \ldots, x_n)$
  - $x_i = 1$ iff term *i* is present in document *x*.
- **"Independence":** terms occur in documents independently

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:

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

Binary Independence Model

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

Constant for a given query

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 $p_i = p(x_i=1 \mid R,q); \quad r_i = p(x_i=1 \mid NR,q);$

• Assume, for all terms not occurring in the query ($q_i=0$) $p_i = r_i$

Then...

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

Binary Independence Model

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

All matching terms

Non-matching query terms

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

All matching terms

All query terms

Binary Independence Model

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

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

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

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 $(N- n)/n \approx$ 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
   - $r_i = n_i/N$
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!)

$$\text{sim}(d_j,q) \sim \sum_{i=1}^{t} w_{i,q} \quad x \quad w_{i,j} \quad x \quad \left(\log \frac{P(k_i \mid R)}{1-(P(k_i \mid R)} \quad + \quad \log \frac{1-P(k_i \mid \sim R)}{P(k_i \mid R)} \right)$$

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
     - $p_i = |V_i|+0.5 \,/\, (|V|+1)$
   - Assume if not retrieved then not relevant
     - $r_i = (n_i - |V_i| + 0.5) \,/\, (N - |V| + 1)$
4. Go to 2. until converges then return ranking

Contoh

d_i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
x_1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
x_2	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0