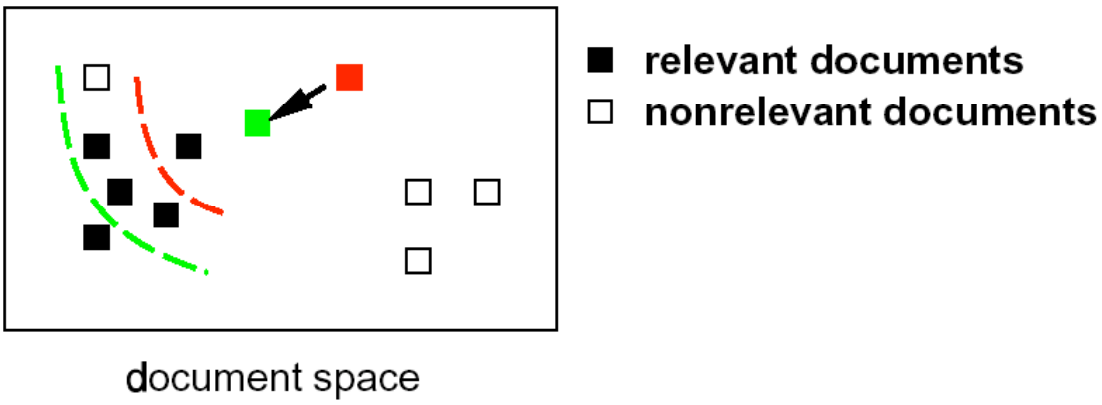
CSA4020

Multimedia Systems:
Adaptive Hypermedia Systems

Lecture 8:
Relevance Feedback

Basic Assumptions

- Similar docs are near each other in vector space
- Starting from some initial query, the query can be reformulated to reflect subjective relevance judgements given by the user
- By reformulating the query we want to move the query closer to more relevant docs and further away from nonrelevant docs
- In statistical model of IR, reformulating query means re-weighting terms in query
- In probabilistic model of IR, query reformulation means adding terms to, or removing terms from, original query



- Not failsafe: may move query towards nonrelevant docs!

The Ideal Query

- If we know the ideal answer set *Rel*, then the ideal query is:

$$Q_{opt} = \frac{1}{R} \sum_{D_i \in Rel} D_i - \frac{1}{N - R} \sum_{D_i \in Nonrel} D_i$$

- In reality, a typical interaction will be:

  User formulates query and submits it
  IR system retrieves set of documents
  User selects *R'* and *N'*

$$Q^{i+1} = \alpha Q^i + \frac{\beta}{|R|} \sum_{D_i \in R} D_i - \frac{\gamma}{|N|} \sum_{D_i \in N} D_i$$

$$Q^{i+1} = \alpha Q^i + \beta \sum_{D_i \in R} D_i - \gamma \sum_{D_i \in N} D_i$$

  where 0 <= α, β, γ<= 1 (and vector magnitude usually dropped...)

Determining the parameters

- What are the values of α, β and γ?

  γ is typically given a value of 0.75, but this can vary. Also, after a number of iterations, the weights of the original terms can be greatly reduced

  If α and β have equal weight, then relevant and nonrelevant docs make equal contribution to reformulated query

  If $\alpha = 1$, $\beta = 0$, then only relevant docs are used in reformulated query

  Usually, use $\alpha = 0.75$, $\beta = 0.25$

Example

(www.cpe.ku.ac.th/~arnon/Mirror/ir-p/Notes/RelFeedback/sld007.htm)

Q: (5, 0, 3, 0, 1)
R: (2, 1, 2, 0, 0) N: (1, 0, 0, 0, 2)
$\alpha = 0.5$, $\beta = 0.25$ ($\gamma = 1.0$)

$$Q' = Q + 0.5R - 0.25N$$
$$= (5, 0, 3, 0, 1) + 0.5(2, 1, 2, 0, 0) - 0.25(1, 0, 0, 0, 2)$$
$$= (5.75, 0.5, 4, 0, 0.5)$$

- How many docs to use in *R'* and *N'*?

  Use all docs selected by user

  Use all rel docs and highest ranking nonrel docs

  Usually, user selects only relevant docs...

- Should entire document vector be used?

  Really want to identify the *significant* terms...

  Use terms with high-frequency/weight

  Use terms in doc adjacent to terms from query

  Use only common terms in *R'* (and *N'*)

Query Splitting

- Query reformulation attempts to modify query so that it moves closer to relevant docs

- User is making relevance judgements about docs in the retrieved set...

- ... so intention is to make new query similar to docs in relevant set

- What happens when the relevant set contains documents which do not cluster, or which form more than one cluster?

  Nothing can be done if there is no cluster

  Otherwise, detect the multiple clusters and reformulate query for each

  Results then need to be merged

Document Space Modification

- Query reformulation assumes that the user has problems describing the ideal document with the correct terminology
- Also assumes that document description is correct
- Alternative is to assume that user query is correct description of relevant document, and the document space is incorrectly described
- When a user gives feedback, modify documents to make them more, or less, like the query
- Document vectors are changed permanently: make changes small enough so that several repetitions are required to make a big change
- Nonrelevant documents tend to cluster
- Results are not repeatable, so evaluation is hard

- Query splitting and document space modification are rarely used

  Query splitting: requires user to mark large number of documents as relevant

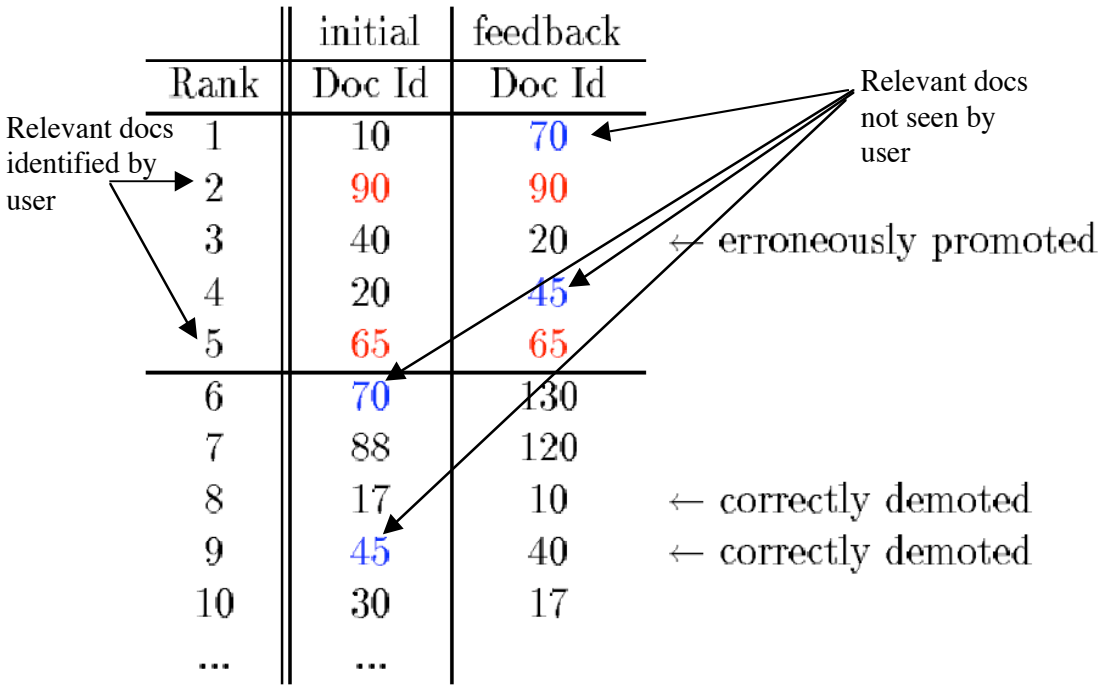
  Document space modification: expensive and inconclusive

Evaluating Relevance Feedback Methods

- RF should select more relevant and previously unseen documents

  Higher ranking of previously judged to be relevant documents not a good indication of RF performance

Partial Rank Freezing



	initial	feedback
Rank	Doc Id	Doc Id
1	10	70
2	90	90
3	40	20
4	20	45
5	65	65
6	70	130
7	88	120
8	17	10
9	45	40
10	30	17
...	...	

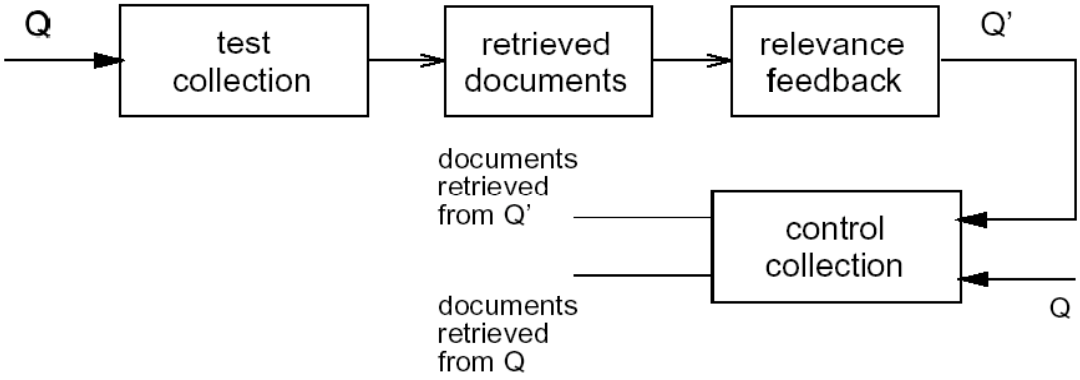
- Retrieved docs marked as relevant are frozen at their current rank. All others are re-ranked
- Only top 5 docs are returned to user

Pruning

- Retrieve docs as usual in feedback process
- Remove those previously marked as relevant (also user has already seen these...)
- Move up the remaining docs
- Evaluate precision and recall
- Takes into account the effect on relevant docs used in feedback process
- Partial Rank Freezing can create abnormal behaviour in the evaluation process

Test and Control Collections

- Split doc collections into 2 sets
- Derive modified query *Q'* from the test collection
- Apply reformulated query *Q'* to the control collection



Probabilistic Relevance Feedback

- Want to add more terms to the query so the query will resemble documents marked as relevant

- How do we select which terms to add to the query?

  Rank terms in marked documents and add the first **m** terms

$$w_i = \log \frac{r_i \left(N - n_i - R + r_i \right)}{\left(R - r_i \right)\left(n_i - r_i \right)}$$

  where:

  N: no. of docs in the collection
  n_i: document frequency of term *i*
  R: no. of relevant docs selected
  r_i: no. of docs in R containing term *i*

- Compares frequency of occurrence of term in R with document frequency