relevance feedback

ISU535 05X2

virgil pavlu

many slides courtesy James Allan@umass

relevance feedback

Observations:

• A Query only approximates an information need

• Users often start with short queries (poor approximations)

• *People* can improve queries after seeing relevant and non-relevant documents

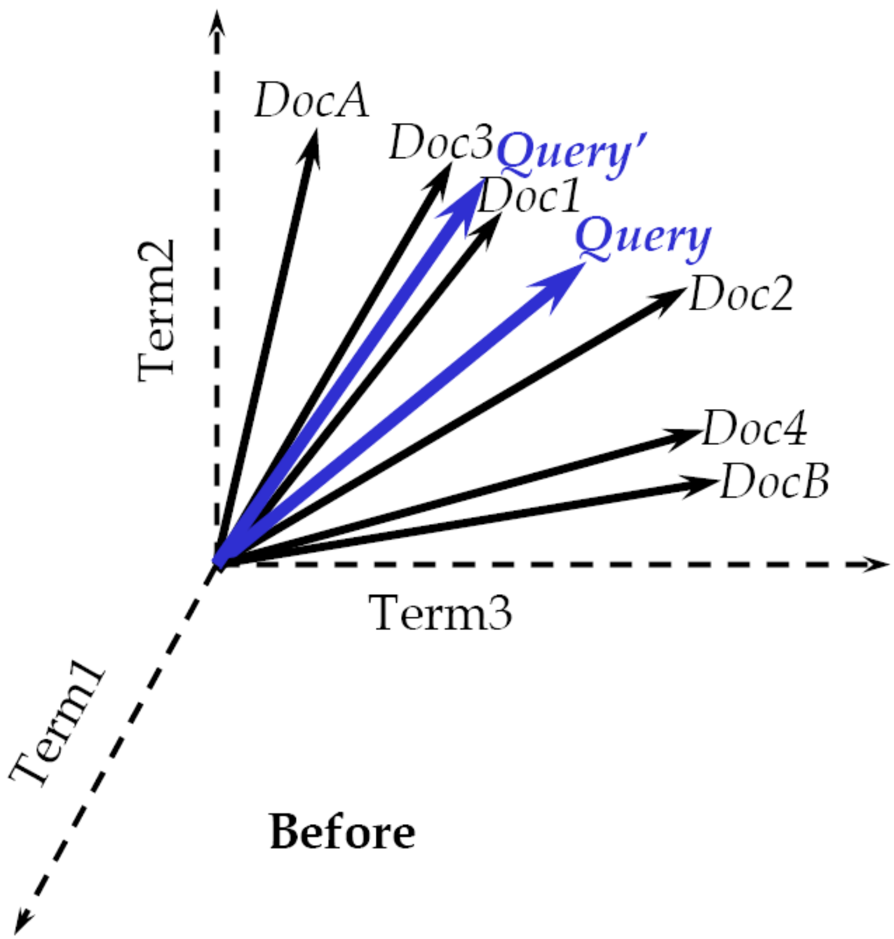
- by adding and removing terms
- by reweighting terms
- by adding structure (AND, OR, NOT, PHRASE, etc)

Question: Can a better query be created *automatically* by analyzing relevant and nonrelevant documents?

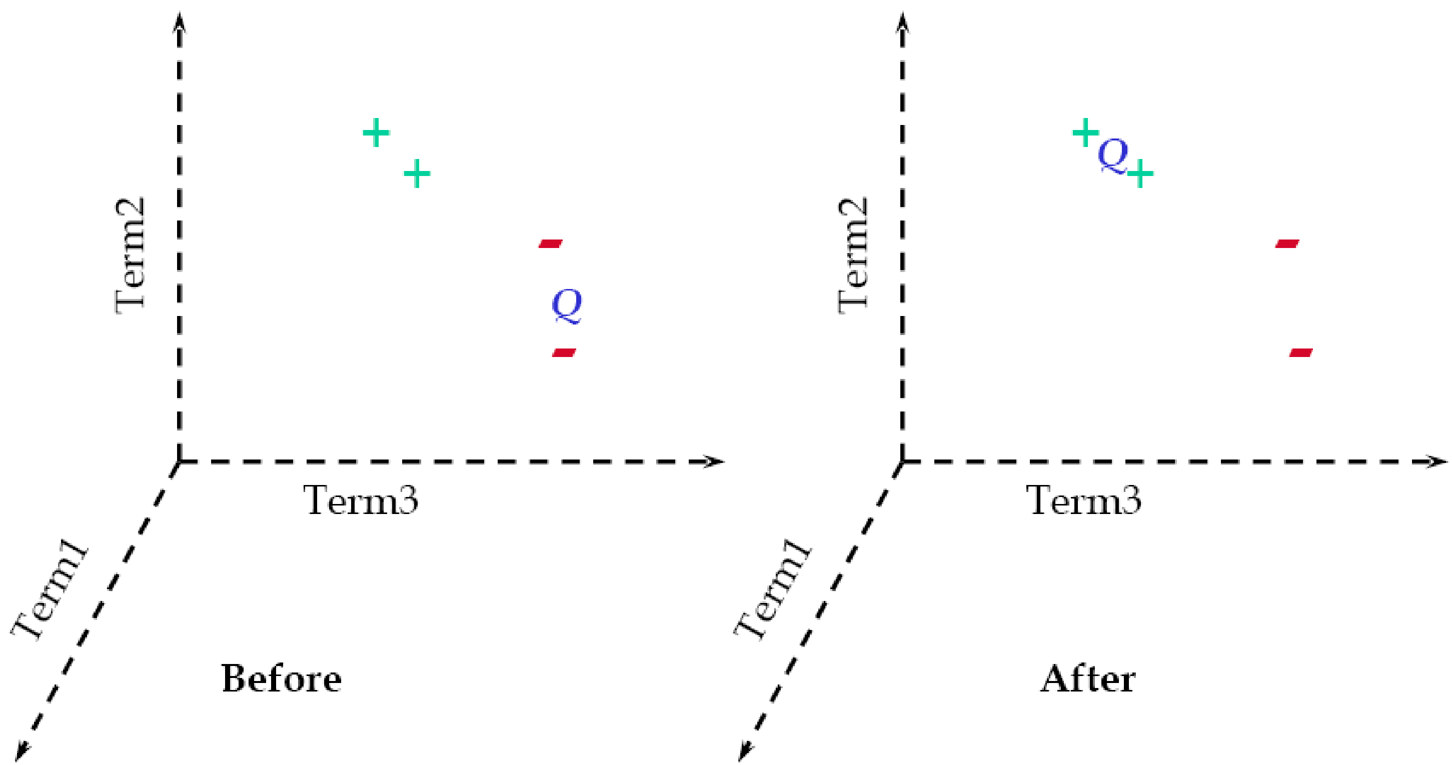
relevance feedback

- "Real" relevance feedback
  - System returns results
  - User provides some feedback
  - System returns different—better, we hope—results
- "Assumed" relevance feedback
  - System gets results but does not return them
  - Uses returned results to "guess" what was probably meant
  - Modifies query without supervision
  - System returns enhanced—and we hope better—result list
- Occurs in different models
  - Vector space is used most often (we'll focus on it)
  - Language modeling
- Good success with "assumed" relevance (relevance models)
- Less obviously good results for "real" feedback

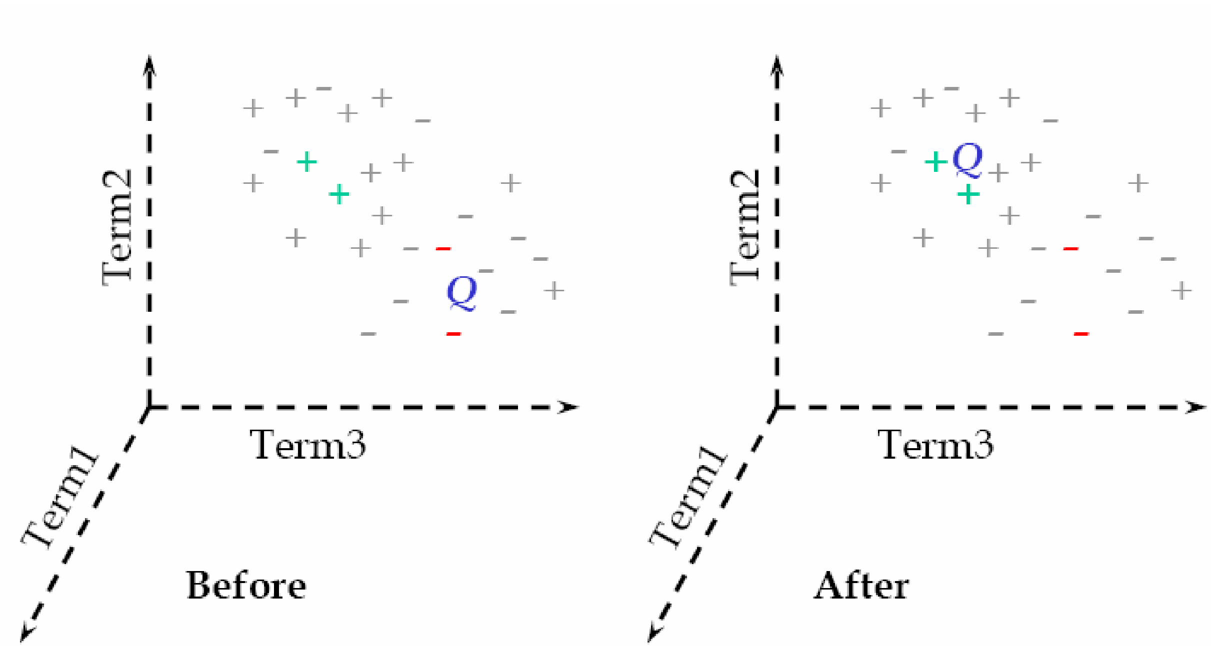
relevance feedback in the vector space



relevance feedback in the vector space



relevance feedback in the vector space



How can relevance feedback save time if a person has to read documents?

relevance feedback

Hypothesis: A better query can be created *automatically* by analyzing relevant and non-relevant documents

• Relevant passages and phrases can also be identified, but this is not common

• Assumes relevant and non-relevant documents are easy for people to identify

• Can be viewed as a form of "query-by-example"

•Common Simplifying Assumptions:

- Unstructured query (terms and weights, but no operators)
- Binary relevance judgements (relevant, not relevant)

relevance feedback in the vector space

- **Goal**: Move new query closer to relevant documents
- **Approach**: New query is a weighted average of original query, and relevant and non-relevant document vectors

$$Q' = Q + \alpha \frac{1}{|R|} \sum_{D_j \in R} D_j - \beta \frac{1}{|NR|} \sum_{D_j \in NR} D_j$$

Often written
$Q' = \alpha Q + \beta R - \gamma N$

where α and β are constants that represent the relative importance of positive and negative feedback

Variations:

- Different values of α and β
- Vector length (number of terms added to the query)
- Which documents are used for training
  – all, best, uncertain, etc

relevance feedback in the vector space

$$Q' = Q + \alpha \frac{1}{|R|} \sum_{D_j \in R} D_j - \beta \frac{1}{|NR|} \sum_{D_j \in NR} D_j$$

Original Query: **(5, 0, 3, 0, 1)**
Document D1, Relevant:
(2, 1, 2, 0, 0)
Document D2, Non-relevant:
(1, 0, 0, 0, 2)

α = **0.50**, β = **0.25**

Q'**=** Q **+ 0.5** D1 **- 0.25** D2
= (5, 0, 3, 0, 1) + 0.5 (2, 1, 2, 0, 0) - 0.25 (1, 0, 0, 0, 2)
= (5.75, 0.50, 4.00, 0.0, 0.5)

example

Original TREC Topic:

<num> Number: 106
<dom> Domain: Law and Government
<title> Topic: U.S. Control of Insider Trading
<desc> Description:
Document will report proposed or enacted changes to U.S. laws and regulations designed to prevent insider trading.
<con> Concept(s):
1. insider trading
2. securities law, bill, legislation, regulation, rule
3. Insider Trading Sanctions Act, Insider Trading and Securities Fraud Enforcement Act
4. Securities and Exchange Commission, SEC, Commodity Futures Trading Commission, CFTC, National Association of Securities Dealers, NASD
<fac> Factor(s):
<nat> Nationality: U.S.

example: query processing (INQUERY)

Automatically processed query:

```
#WSUM ( 1.0
!Terms from <title> field:
2.0 #UW50  (  Control of Insider Trading )
2.0 #PHRASE ( #USA Control ) 5.0 #PHRASE ( Insider Trading )
!  Terms from <con> field:
2.0 #PHRASE( securities law) 2.0 bill 2.0 legislation 2.0 regulation
2.0 rule 2.0 #3( Insider Trading Sanctions Act)
2.0 #3( Insider Trading and Securities Fraud Enforcement Act )
2.0 #3( Securities and Exchange Commission) 2.0 SEC
2.0 #3(Commodity Futures Trading Commission) 2.0 CFTC
2.0 #3( National Association of Securities Dealers) 2.0 NASD
!  Terms from <desc> field:
1.0 proposed 1.0 enacted 1.0 changes 1.0 #PHRASE ( #USA laws )
1.0 regulations 1.0 designed 1.0 prevent
2.0 #NOT(#FOREIGNCOUNTRY)  )
```

example: relevance feedback added

Automatically modified query, top 10 documents judged:

#WSUM (1
3.882349 #UW50(control inside trade) 2.208832 #SUM(#usa control)
145.571381 #SUM(inside trade) 22.084291 #SUM(secure law)
22.693285 bill 20.984898 legislate 10.354733 regulate
6.540223 rule 1.529766 #OD3(inside trade sanction act)
3.290401 #OD4(inside trade secure fraud enforcement act)
4.8404 #OD4(secure exchange commission) 43.578438 sec
0.94752 #OD3(commodity future trade commission) 1.074666 cftc
2.864415 #OD4(national associate secure deal) 21.846081 nasd
0.542252 propose 2.45709 enact 0.988893 change 4.354009 #SUM(#usa law)
0.799089 design 1.727937 prevent 0.346877 #NOT(#foreigncountry)
4.599784 *drexel* 2.052418 *fine* 1.845434 *subcommittee*
1.69074 *surveillance* 1.597542 *markey* 1.528179 *senate*
1.186563 *manipulate* 1.101982 *pass* 1.060453 *scandal*
0.921561 *edward*)

relevance feedback in the vector space

Term Selection:
- None (original query terms, only)
- All terms
- Most common terms
- Most highly weighted terms

Weighting:

- **Ide** : $\alpha=1$, $\beta=1$, don't normalize by number of judged documents

- **Ide Dec Hi**: $\alpha=1$, $\beta=1$, use only the highest ranked non-relevant document(s), don't normalize by number of judged documents

- **Rocchio**: Choose α and β such that $\alpha>\beta$ and $\alpha+\beta=1$

$$Q' = Q + \alpha \frac{1}{|R|} \sum_{D_j \in R} D_j - \beta \frac{1}{|NR|} \sum_{D_j \in NR} D_j$$

relevance feedback in the vector space

- Ide Dec Hi is effective when there are *a few* judged documents

- Rocchio (α=0.75, β=0.25) is effective when there are *many* judged documents

- Expanding by *all* terms is best, but selecting *most common* terms also works well
  - Depends somewhat on the retrieval model

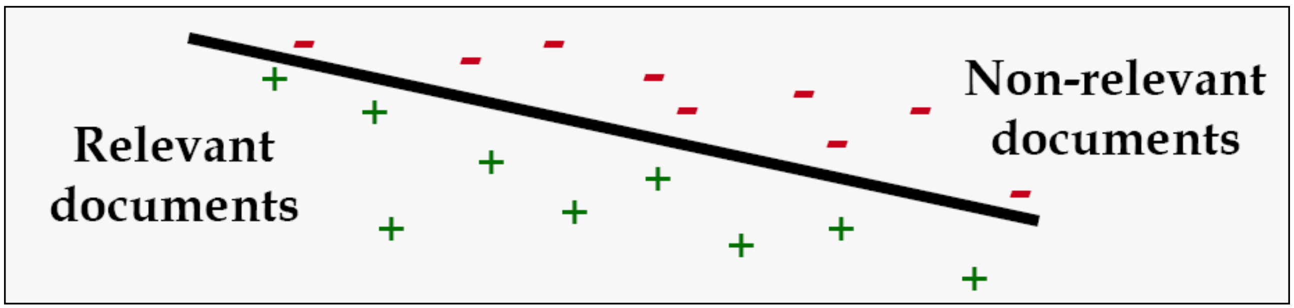
- Coping with negatively weighted terms
  - Vector space does not allow negative weights for cosine similarity
  - Usually drop terms that end up negatively weighted
  - Can create a “not like this” vector consisting of negative terms

- Difficult to balance issues correctly

relevance feedback : ML

- An unstructured vector query is a linear discriminator
  - e.g., $w_1 * t_1 + w_2 * t_2 + \ldots + w_n * t_n$

- The goal is to learn weights that separate the relevant documents from the non-relevant documents



- If the documents are *linearly separable*, a learning algorithm can be chosen that is guaranteed to converge to an optimal query
- If the documents are not linearly separable, a learning algorithm can be chosen that minimizes the total amount of error

relevance feedback : ML

- Unstructured queries:
  - Perceptron algorithm (Rocchio)
  - EG (a form of Perceptron algorithm)
  - Regression
  - Neural network algorithms
  - SVM
  - : : :
- Structured queries
  - Decision trees
  - Neural network algorithms
  - Sleeping Experts
  - Ripper
  - : : :

rocchio and the perceptron

- The Rocchio relevance feedback algorithm is similar to the fixed increment version of the Perceptron rule:

$$\vec{Q}' = \vec{Q} \pm c\vec{D}_i \quad \begin{cases} + \ \text{if } D_i \in R \\ - \ \text{if } D_i \in \overline{R} \end{cases}$$

- The Perceptron:
  - requires *repeated* exposure to training data,
  - requires random sampling,
  - works best if R and NR are of similar size, and
  - is optimal if R and NR can be separated by a hyperplane (otherwise it oscillates).

relevance feedback: adding structure

Basic Process:

- Generate candidate operators (Boolean,Phrase, proximity, etc)
  - algorithms: exhaustive, greedy/selective
- Add some or all candidates to document representations
- Weight like other terms

Effectiveness:

- Extremely effective for proximity operators
- Boolean?

relevance feedback

- Relevance Feedback could also modify *document* representation
  - document space modification
  - connectionist learning (changing weights in network)

- Assumptions:
  - a person's relevance judgements are consistent
  - modifications for one person are meaningful for another

- Never shown to be effective *consistently*

- An old idea, periodically resurfaces
  - recommender systems

- Difficult to figure out how searchers should use it

summary (halfway)

- Relevance feedback can be *very* effective

- Effectiveness depends on number of judged documents

- Significantly outperforms best human queries, given enough judged documents

- Results can be unpredictable with less than five judged documents

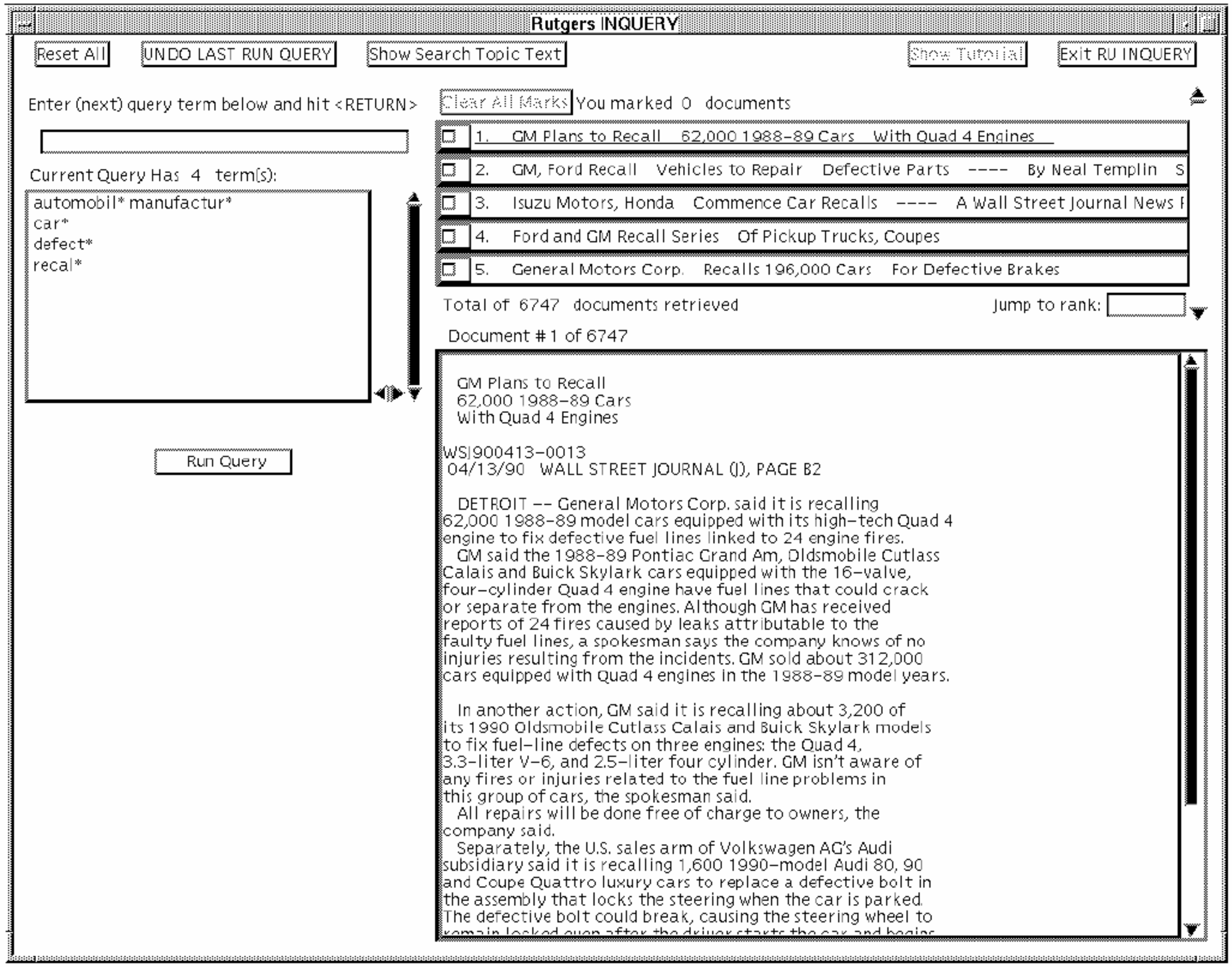
- Not used often in production systems, e.g., Web
  - consistent mediocre performance preferred to inconsistently good/great results
  - Stick with “documents like this one” variant

- An area of very active research (many open questions)

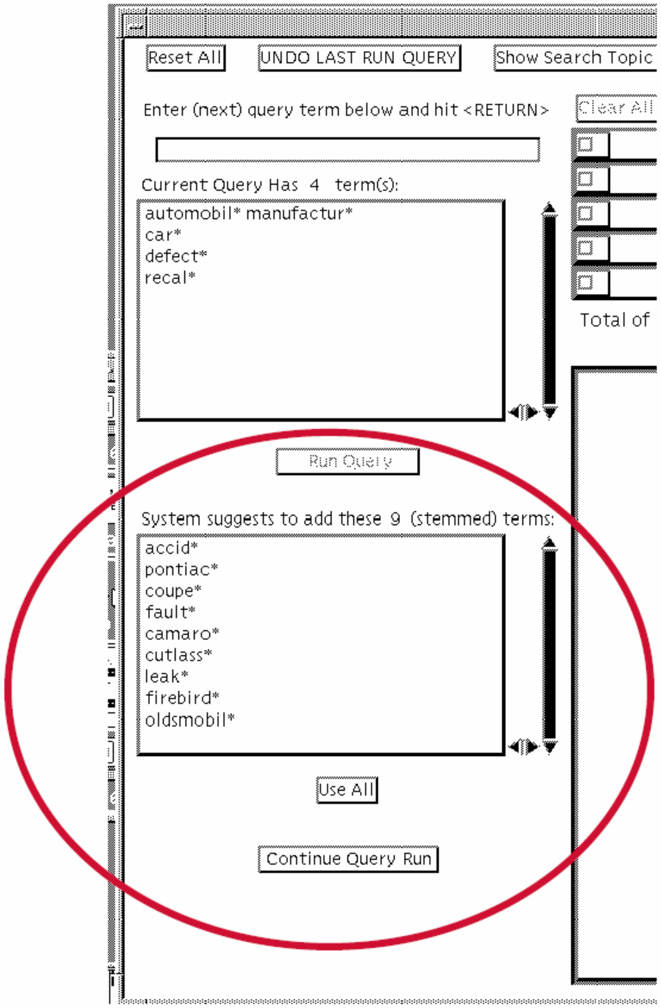
using relevance feedback

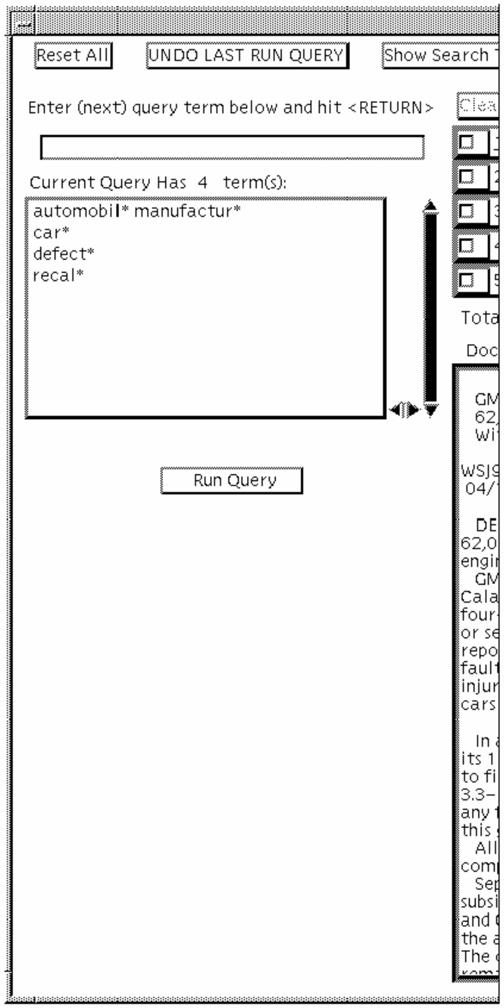
- Relevance feedback is not widely used
- Few studies explore the user side of feedback
  - Don't necessarily answer that question, but are still interesting
- Jürgen Koenemann and Nick Belkin looked at this
  - "A case for interaction: A Study of Interactive Information Retrieval Behavior and Effectiveness", CHI 1996
- User study of 64 users
- Presented with three styles of relevance feedback
  - Opaque, relevance feedback is "magic" behind the scenes
  - Transparent, same as *opaque* but users shown expansion terms
  - Penetrable, user given chance to edit list of terms before re-run

base system used



allowing user access

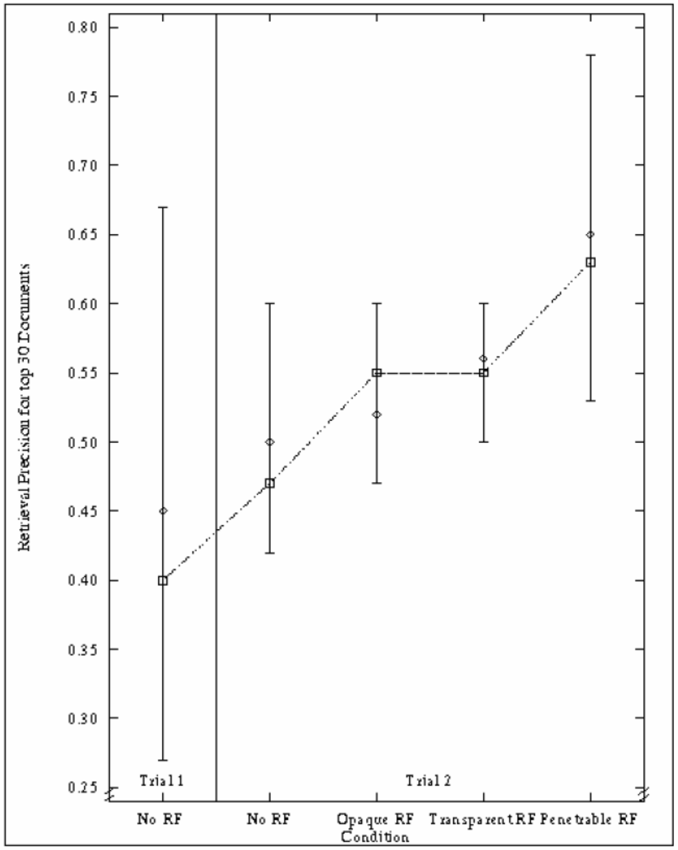




interface experiment

- Two query construction approaches
  - First without relevance feedback
  - Second with one of three RF approaches (randomly assigned)
- Task is to construct a good long-term query
- Evaluation is based on effectiveness of final query
- No difference between users on first task

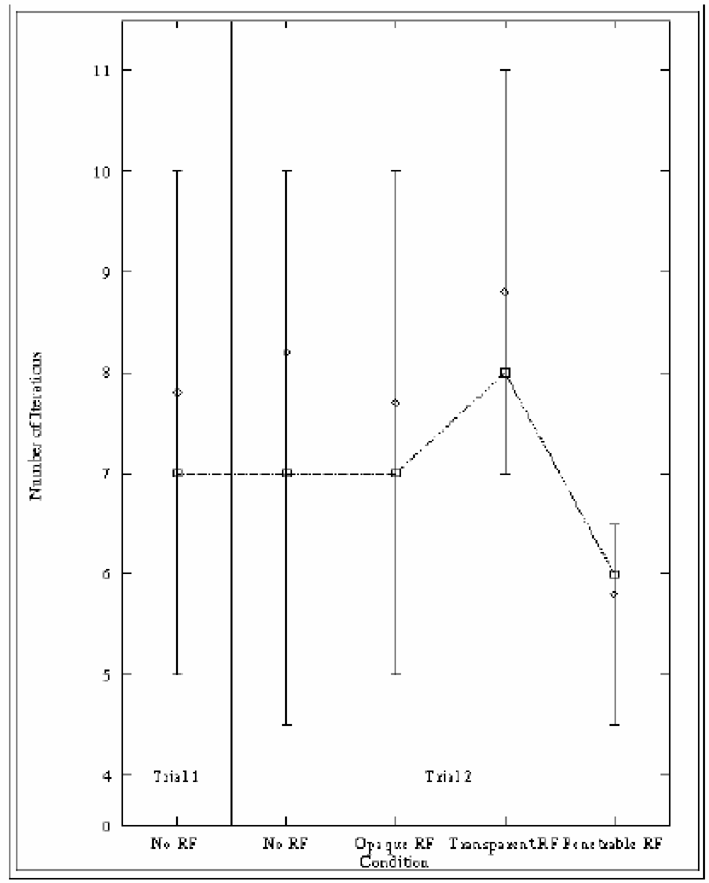
feedback effectiveness



- Precision at 30 documents
- Clear improvements from use of RF
- Opaque and transparent the same (by design)
- Penetrable best
- Only statistically significant difference is between penetrable and base
- Results comparable for precision at 100documents

feedback: behavior

- Task was to build a good query
- How many attempts do people make?
- For some reason, transparent interface encouraged an extra iteration
- Penetrable interface took one less than “normal”
- Not clear what this means



was feedback used by searcher?

- Where did words they chose come from?
  - Copied from lists provided by feedback
  - Added automatically by system
- Users typed short queries
- Feedback added many terms
- Penetrable system encouraged fewer terms
  - But resulted in more effective queries (faster)

Mean Number & Sources of Query Terms

Relevance Feedback Condition	User Controlled: User Typed	User Controlled: Copy from RF	User Controlled: Σ	Added by RF SYS	Σ
Topic 162:					
None	6.9	n/a	6.9	n/a	6.9
Opaque	10.9	n/a	10.9	35.9	46.8
Transparent	3.3	9.1	12.4	42.8	55.1
Penetrable	6.3	24.4	30.6	n/a	30.6
Topic 165:					
None	6.0	n/a	6.0	n/a	6.0
Opaque	3.8	n/a	3.8	20.5	24.3
Transparent	4.3	5.3	9.5	17.8	27.3
Penetrable	3.3	9.5	12.8	n/a	12.8
162&165:					
None	6.4	n/a	6.4	n/a	6.4
Opaque	7.3	n/a	7.3	28.2	35.5
Transparent	3.8	7.2	10.9	30.3	41.2
Penetrable	4.8	16.9	21.7	n/a	21.7

subjective reactions

- Subjects “liked” the penetrable version

- Subjects in opaque condition expressed desire to “see and control” what happened

- Subjects comments that feedback made them “lazy”
  - Task of generating terms changed to task of selecting terms

relevance feedback: assumed

- True relevance feedback is supervised
  - Feedback is done based on *genuine* user annotations
- What happens if we try to guess what is relevant?
- Assume many top ranked documents are relevant
  - Optionally find a collection of probably non-relevant documents
- Modify query on that assumption
- Re-run that new query and show results to user
- What happens?
- Pseudo-relevance feedback
  - Blind relevance feedback
  - Local feedback
  - …

Local Context Analysis

- Assumed relevance feedback

- Observations
  - Existing techniques improved queries on average
  - But some queries had serious drop in effectiveness
  - Top ranked documents were not always right
  - Often caused by match of a single query word
  - Not every word is useful to add to queries

- Inspired creation of LCA

- Major focus is on getting better terms for expansion
  - Finding terms to consider
  - Selection of terms
  - Weighting of selected terms

selecting candidate terms

- Run query to retrieve passages
  - Similar to most "assumed" relevance work
  - Passage-retrieval unique to LCA (at the time)
  - Uses 300-word passages
- Select expansion concepts from retrieved set
- Why passages?
  - Minimizes spurious concepts that occur in lengthy documents

selecting candidate terms

- Parse document collection
- Generate part of speech tagging

– The/AT bill/NN has/HVZ been/BEN reworked/VBN since/CS it/PPS was/BEDZ introduced/VBN ,/, in/IN order/NN to/TO meet/VB some/DTI employer/NN objections/NNS ./. But/CC the/AT measure/NN still/RB is/BEZ opposed/VBN by/IN the/AT construction/NN industry/NN ,/, which/WDT argues/VBZ that/CS it/PPS would/MD impose/VB unionism/NN and/CC higher/JJ costs/NNS on/IN much/AP of/IN the/AT industry/NN 's/$ work/NN ./.

- Select only noun phrases
  - Shown to be critical in most retrieval systems
  - Generally particularly useful for expansion
  - Could easily be extended if useful
- Adjective-noun phrases, verbs, …
  - Note that tagging is automated, so makes mistakes!

weighting terms

- Want "concepts" that occur near query words
  - The more query words they occur near, the better
  - Count co-occurrences in 300-word windows of text (passages)
- To avoid coincidental co-occurrence in a large document
- Uses the following ad-hoc function to weight concepts

$$
\begin{aligned}
f(c,Q) &= \prod_{w_i \in Q} \left(0.01 + \text{co\_degree}(c, w_i)\right)^{idf(w_i)} \\
\text{co\_degree}(c,w) &= \max\left(\frac{n_{cw} - En(c,w) - 1}{n_c}, 0\right) \\
En(c,w) &= \frac{n_w n_c}{N} \\
idf(w) &= \min(1.0, \log_{10}(N/n_w)/5)
\end{aligned}
$$

Importance of word

Measure co-occurrence

Floor the IDF component

Slow its growth

using expanded query

- Developed using Inquery
- Incorporate using weighted sum
  - Weight original query and expansion query equally

$$
\begin{aligned}
Q_{new} = \#wsum(1.0\ 1.0\ & Q_{original} \\
& 1.0\ Q_{lca})
\end{aligned}
$$

$$
Q_{lca} = \#wsum(1.0\ 1.0\ c_1\ 1.0\ c_2\ \ldots\ 1.0\ c_{30})
$$

- Variations
  - Lower weight on each subsequent term
    - More important the more terms that are added
  - Weight original query equally with a single expansion concept
    - Only works when query is not very reliable

example

- TREC query 213

– *As a result of DNA testing, are more defendants being absolved or convicted of crimes?*

- Expansion concepts

dna-pattern	dna-testing	lifecodes	dna-test-results
dna-lab	dna-evidence	dna-test	dna-profile
defense-attorneys-challenging-reliability			bureau-expert
lawyer-peter-neufeld		new-york-city-murder-case	
michael-baird	procedures-track-record		dna-laboratory
oregon-rape-case	mark-storolow	laboratory-geletin	
supermarket-merchandise		thomas-caskey	procedures-lifecode
lifecodes-corp	tests-reliability	maine-case	rape-conviction
dna-strand			

summary

- Relevance feedback
  - Real or assumed
- Real relevance feedback
  - Usually improves effectiveness significantly
  - Not always stable with very few documents judged
  - Difficult to incorporate into a usable system
  - "Documents like this one" is a simple instance
- Assumed relevance feedback
  - Also called "pseudo relevance feedback" or "local feedback"
- Or "quasi-relevance feedback" or …
  - Rocchio-based approaches effective but unstable
  - LCA comparably effective (maybe better) but more stable
  - Relevance models provide formal framework