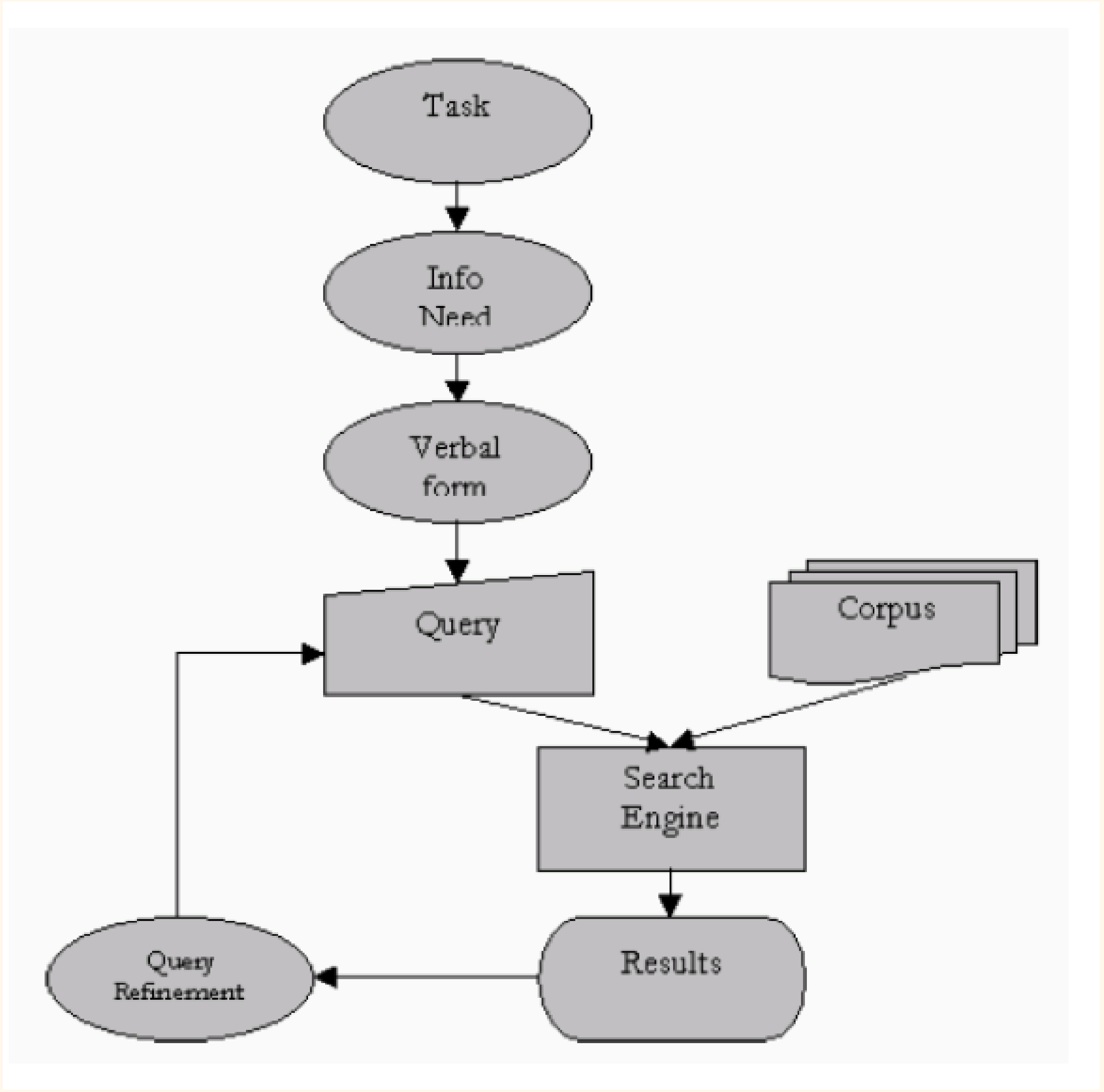
Relevance Feedback

- Concepts
- Global methods
  - Query expansion
    - Thesauri
- Local methods
  - Rocchio's algorithm
  - Probabilistic formulations
  - Implicit/Blind Relevance Feedback
- Papers

Relevance Feedback



Recall that the information-seeking process is iterative...

Recall that the information-seeking process is iterative…

Why?

Users might:

- Not know how to optimally word a query…
- Not be sure what they're looking for, but will "know it when I see it"…
- Not be sure what the collection contains…

All of these forces drive the need to iterate!

Remember: "recognition over recall"!

Classical relevance feedback flow:

1. User issues query;
2. System returns initial set of results;
3. User marks some number of documents as +/- relevant
4. System uses this data to compute a better representation of the user's information need
5. Repeat from step 2.

There are three major ways of doing relevance feedback:

1. Explicit Feedback

Users explicitly mark results as +/- relevant.

2. Implicit Feedback

The system attempts to infer +/- relevance from observable user behavior

3. Blind Feedback

The system attempts to infer +/- relevance blindly (no evidence)

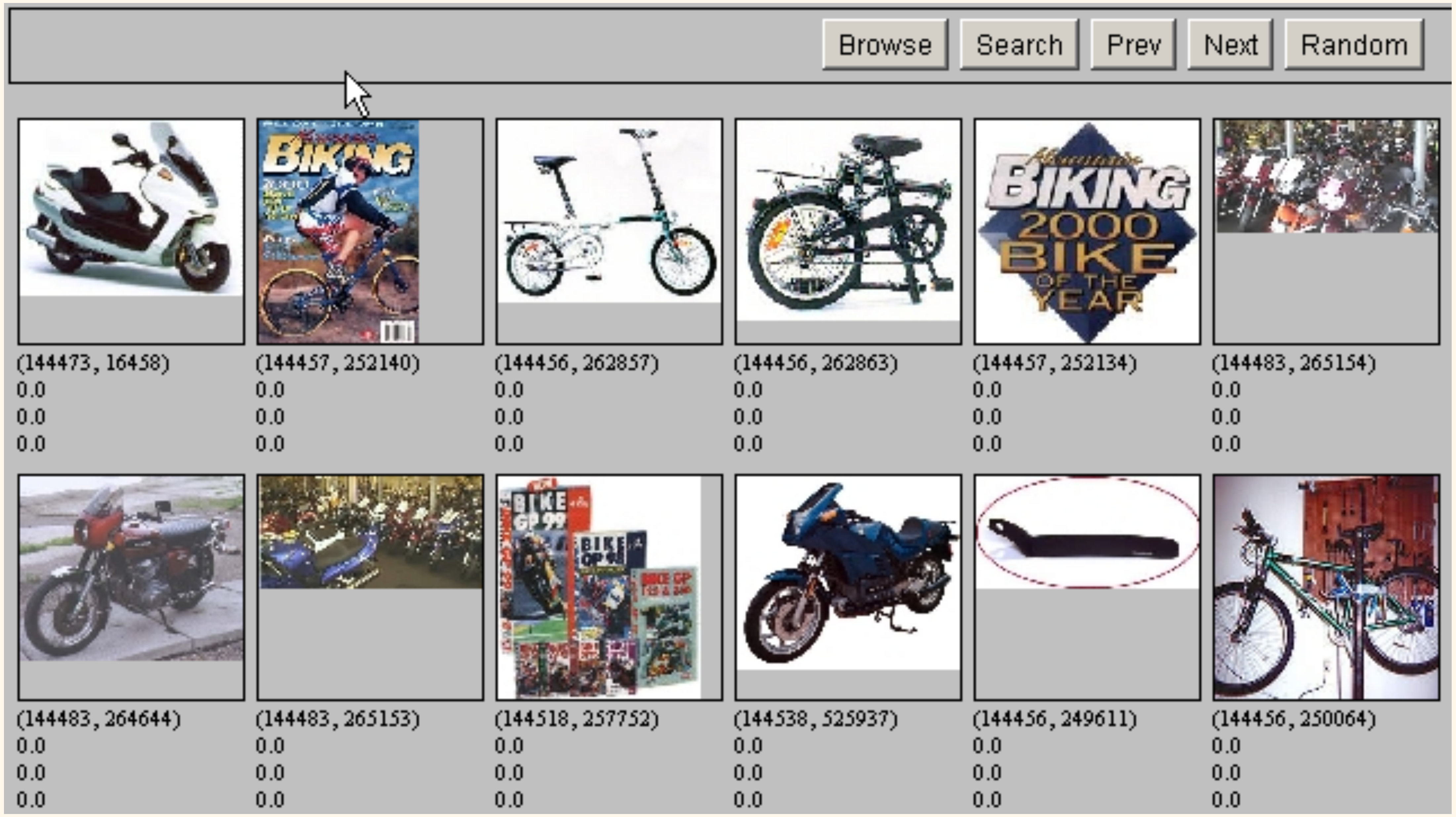
Another way to conceptualize relevance feedback:

"Global" methods adjust a query independent of its results;

Thesaurus-based query expansion, etc.

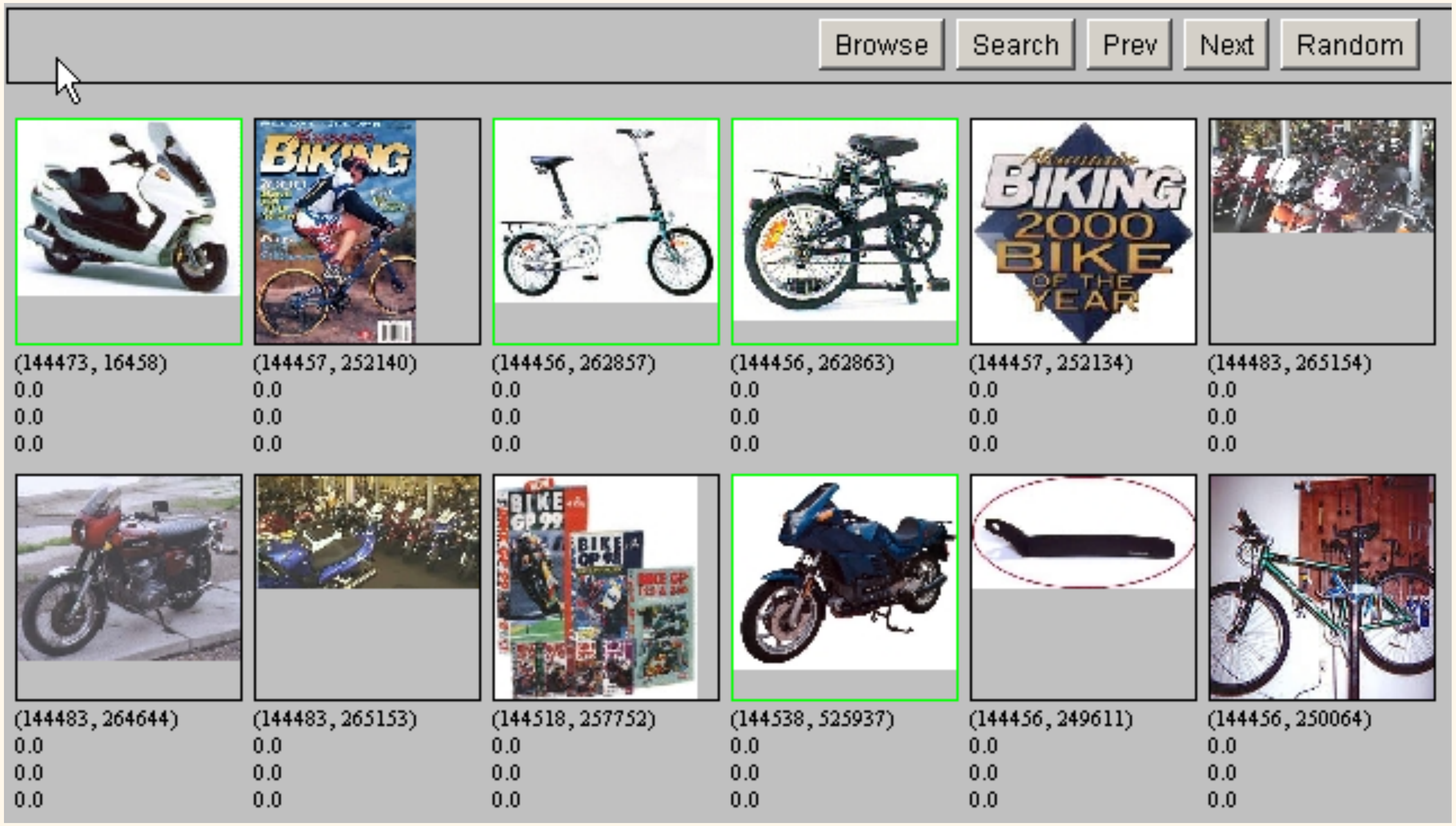
"Local" methods adjust a query relative to its results.

Explicit/pseudo relevance feedback, etc.



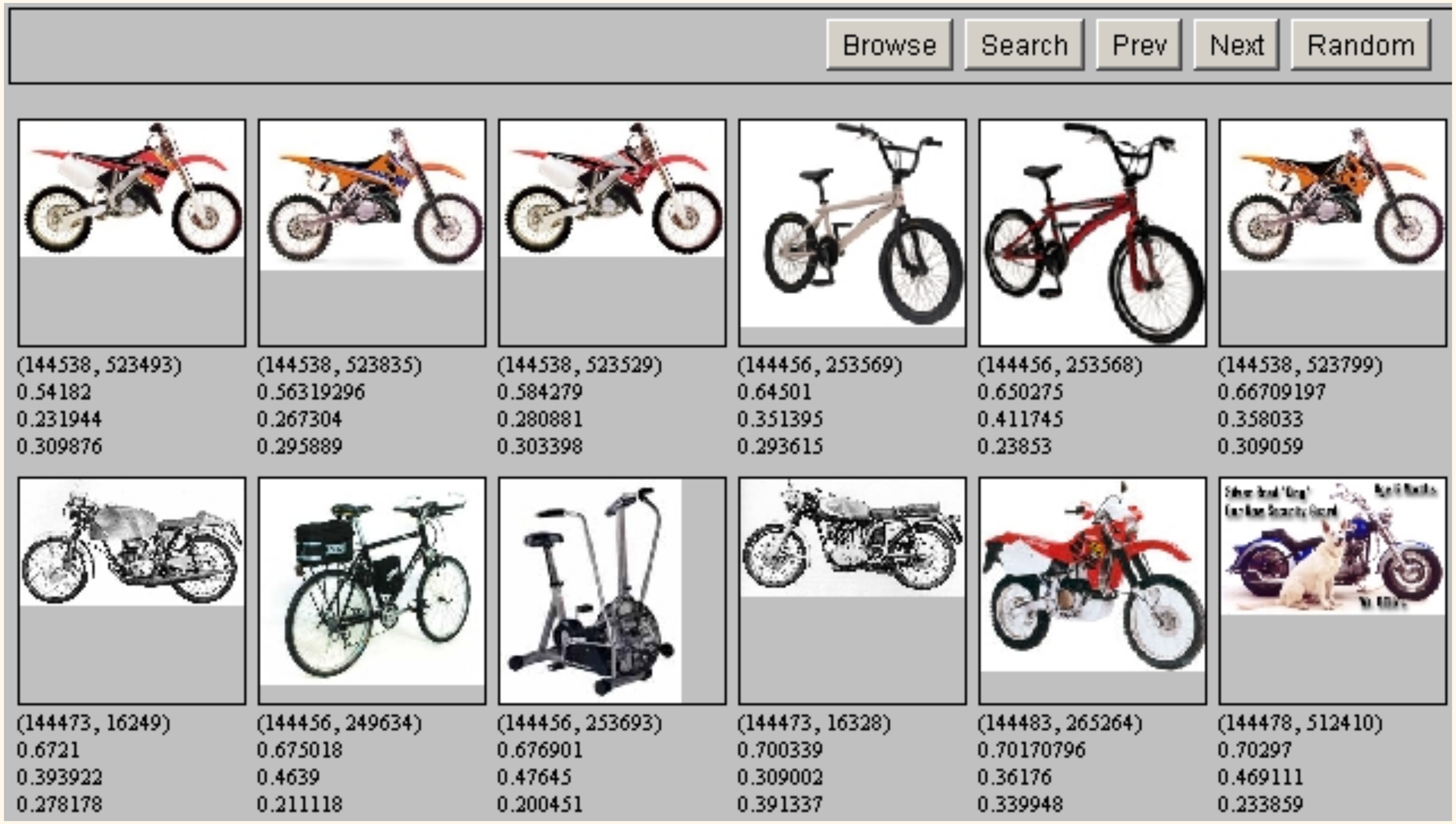
```
(144473, 16458)   (144457, 252140)   (144456, 262857)   (144456, 262863)   (144457, 252134)   (144483, 265154)
0.0               0.0                0.0                0.0                0.0                0.0
0.0               0.0                0.0                0.0                0.0                0.0
0.0               0.0                0.0                0.0                0.0                0.0

(144483, 264644)  (144483, 265153)   (144518, 257752)   (144538, 525937)   (144456, 249611)   (144456, 250064)
0.0               0.0                0.0                0.0                0.0                0.0
0.0               0.0                0.0                0.0                0.0                0.0
0.0               0.0                0.0                0.0                0.0                0.0
```



```
(144473, 16458)  (144457, 252140)  (144456, 262857)  (144456, 262863)  (144457, 252134)  (144483, 265154)
0.0              0.0               0.0               0.0               0.0               0.0
0.0              0.0               0.0               0.0               0.0               0.0
0.0              0.0               0.0               0.0               0.0               0.0

(144483, 264644) (144483, 265153)  (144518, 257752)  (144538, 525937)  (144456, 249611)  (144456, 250064)
0.0              0.0               0.0               0.0               0.0               0.0
0.0              0.0               0.0               0.0               0.0               0.0
0.0              0.0               0.0               0.0               0.0               0.0
```



```
(144538, 523493)  (144538, 523835)  (144538, 523529)  (144456, 253569)  (144456, 253568)  (144538, 523799)
0.54182           0.56319296        0.584279          0.64501           0.650275          0.66709197
0.231944          0.267304          0.280881          0.351395          0.411745          0.358033
0.309876          0.295889          0.303398          0.293615          0.23853           0.309059

(144473, 16249)   (144456, 249634)  (144456, 253693)  (144473, 16328)   (144483, 265264)  (144478, 512410)
0.6721            0.675018          0.676901          0.700339          0.70170796        0.70297
0.393922          0.4639            0.47645           0.309002          0.36176           0.469111
0.278178          0.211118          0.200451          0.391337          0.339948          0.233859
```

Another way to conceptualize relevance feedback:

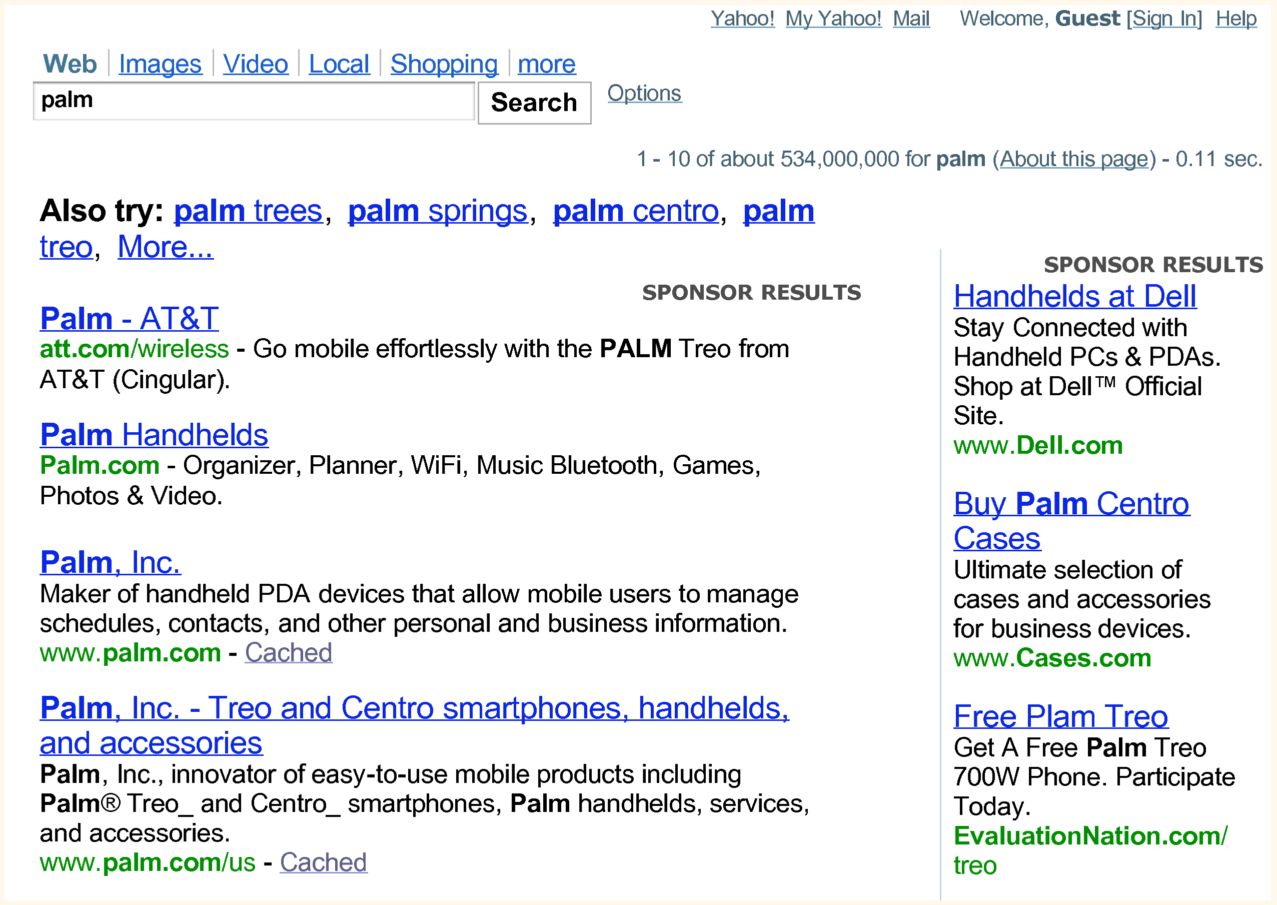
Let's hypothesize an optimal query to represent a given information need...

... the goal of relevance feedback is to adjust the user's query to more closely match the optimal query.

We can do this by adding terms or otherwise altering the user's query based on +/- relevant documents.

“Global” methods adjust a query independent of its results.

Query expansion: increase query effectiveness by adding new, hopefully relevant search terms



Yahoo! My Yahoo! Mail Welcome, **Guest** [Sign In] Help

Web | Images | Video | Local | Shopping | more

palm Search Options

1 - 10 of about 534,000,000 for **palm** (About this page) - 0.11 sec.

Also try: **palm** trees, **palm** springs, **palm** centro, **palm** treo, More...

SPONSOR RESULTS

Palm - AT&T
att.com/wireless - Go mobile effortlessly with the **PALM** Treo from AT&T (Cingular).

Palm Handhelds
Palm.com - Organizer, Planner, WiFi, Music Bluetooth, Games, Photos & Video.

Palm, Inc.
Maker of handheld PDA devices that allow mobile users to manage schedules, contacts, and other personal and business information.
www.**palm**.com - Cached

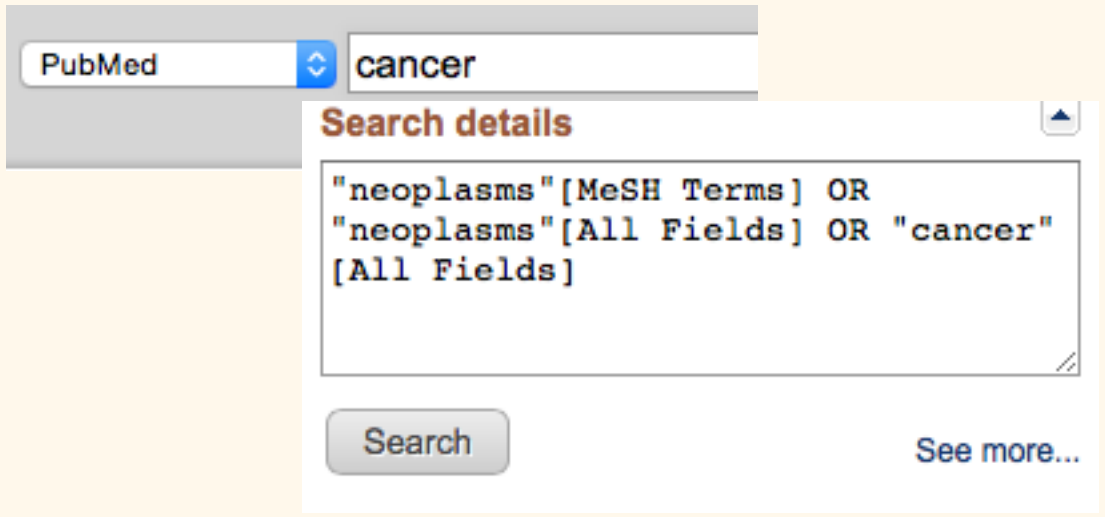
Palm, Inc. - Treo and Centro smartphones, handhelds, and accessories
Palm, Inc., innovator of easy-to-use mobile products including **Palm**® Treo_ and Centro_ smartphones, **Palm** handhelds, services, and accessories.
www.**palm**.com/us - Cached

SPONSOR RESULTS

Handhelds at Dell
Stay Connected with Handheld PCs & PDAs. Shop at Dell™ Official Site.
www.**Dell**.com

Buy **Palm** Centro Cases
Ultimate selection of cases and accessories for business devices.
www.**Cases**.com

Free Plam Treo
Get A Free **Palm** Treo 700W Phone. Participate Today.
EvaluationNation.com/treo

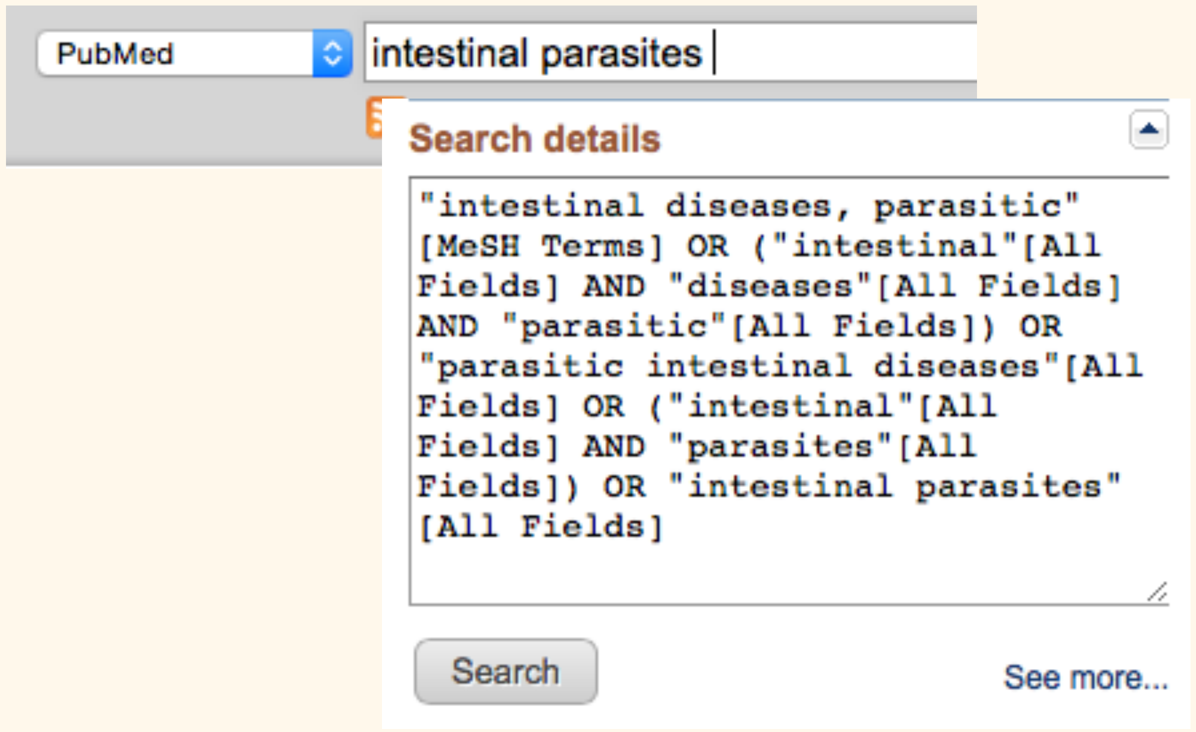


PubMed cancer

Search details

"neoplasms"[MeSH Terms] OR "neoplasms"[All Fields] OR "cancer"[All Fields]

Search See more...



PubMed intestinal parasites

Search details

"intestinal diseases, parasitic"[MeSH Terms] OR ("intestinal"[All Fields] AND "diseases"[All Fields] AND "parasitic"[All Fields]) OR "parasitic intestinal diseases"[All Fields] OR ("intestinal"[All Fields] AND "parasites"[All Fields]) OR "intestinal parasites"[All Fields]

Search See more...

“Global” methods adjust a query independent of its results.

Query expansion: increase query effectiveness by adding new, hopefully relevant search terms

Supervised approaches:
  Thesauri, controlled vocabularies, etc.

Unsupervised approaches:
  Corpus-based association mining, vector embeddings, etc.

“Local” methods adjust a query relative to its results.

The classical approach: Rocchio's algorithm

Rocchio's algorithm operates on a vector space model.

The theory: the optimal query vector maximizes similarity with the relevant documents while minimizing similarity with non-relevant documents.

$$\vec{q}_{opt} = \arg\max_{\vec{q}} [\text{sim}(\vec{q}, C_r) - \text{sim}(\vec{q}, C_{nr})]$$

$$\vec{q}_{opt} = \frac{1}{|C_r|} \underbrace{\sum_{\vec{d}_j \in C_r}}_{\text{relevant document vectors}} \vec{d}_j - \frac{1}{|C_{nr}|} \underbrace{\sum_{\vec{d}_j \in C_{nr}}}_{\text{non-relevant document vectors}} \vec{d}_j$$

Of course, we don't know the full set of relevant and non-relevant documents!

The classical approach: Rocchio's algorithm

The theory: the optimal query vector maximizes similarity with the relevant documents while minimizing similarity with non-relevant documents.

$$\vec{q}_{opt} = \arg\max_{\vec{q}} [\text{sim}(\vec{q}, C_r) - \text{sim}(\vec{q}, C_{nr})]$$

$$\vec{q}_{opt} = \frac{1}{|C_r|} \sum_{\vec{d}_j \in C_r} \vec{d}_j - \frac{1}{|C_{nr}|} \sum_{\vec{d}_j \in C_{nr}} \vec{d}_j$$

$$\vec{q}_m = \alpha \vec{q}_0 + \beta \frac{1}{|D_r|} \sum_{\vec{d}_j \in D_r} \vec{d}_j - \gamma \frac{1}{|D_{nr}|} \sum_{\vec{d}_j \in D_{nr}} \vec{d}_j$$

Original query vector

small set of known-relevant docs

small set of known-non-relevant docs

α, β, γ are weights for how much to rely on the various components.

How should they be set?

The classical approach: Rocchio's algorithm

The theory: the optimal query vector maximizes similarity with the relevant documents while minimizing similarity with non-relevant documents.

$$\vec{q}_{opt} = \arg\max_{\vec{q}} [\text{sim}(\vec{q}, C_r) - \text{sim}(\vec{q}, C_{nr})]$$

$$\vec{q}_{opt} = \frac{1}{|C_r|} \sum_{\vec{d}_j \in C_r} \vec{d}_j - \frac{1}{|C_{nr}|} \sum_{\vec{d}_j \in C_{nr}} \vec{d}_j$$

$$\vec{q}_m = \alpha \vec{q}_0 + \beta \frac{1}{|D_r|} \sum_{\vec{d}_j \in \underset{\substack{\uparrow \\ \text{small set of known-relevant docs}}}{D_r}} \vec{d}_j - \gamma \frac{1}{|D_{nr}|} \sum_{\vec{d}_j \in \underset{\substack{\uparrow \\ \text{small set of known-non-relevant docs}}}{D_{nr}}} \vec{d}_j$$

α, β, γ are weights for how much to rely on the various components.

When we have a lot of judged documents, β and γ should be large...

The classical approach: Rocchio's algorithm

The theory: the optimal query vector maximizes similarity with the relevant documents while minimizing similarity with non-relevant documents.

$$\vec{q}_{opt} = \arg\max_{\vec{q}} [\text{sim}(\vec{q}, C_r) - \text{sim}(\vec{q}, C_{nr})]$$

$$\vec{q}_{opt} = \frac{1}{|C_r|} \sum_{\vec{d}_j \in C_r} \vec{d}_j - \frac{1}{|C_{nr}|} \sum_{\vec{d}_j \in C_{nr}} \vec{d}_j$$

$$\vec{q}_m = \alpha \vec{q}_0 + \beta \frac{1}{|D_r|} \sum_{\vec{d}_j \in D_r} \vec{d}_j - \gamma \frac{1}{|D_{nr}|} \sum_{\vec{d}_j \in D_{nr}} \vec{d}_j$$

small set of known-relevant docs   small set of known-non-relevant docs

α, β, γ are weights for how much to rely on the various components.

When we have fewer judged documents, α should dominate...

The classical approach: Rocchio's algorithm

The theory: the optimal query vector maximizes similarity with the relevant documents while minimizing similarity with non-relevant documents.

$$\vec{q}_{opt} = \arg\max_{\vec{q}} [\mathrm{sim}(\vec{q}, C_r) - \mathrm{sim}(\vec{q}, C_{nr})]$$

$$\vec{q}_{opt} = \frac{1}{|C_r|} \sum_{\vec{d}_j \in C_r} \vec{d}_j - \frac{1}{|C_{nr}|} \sum_{\vec{d}_j \in C_{nr}} \vec{d}_j$$

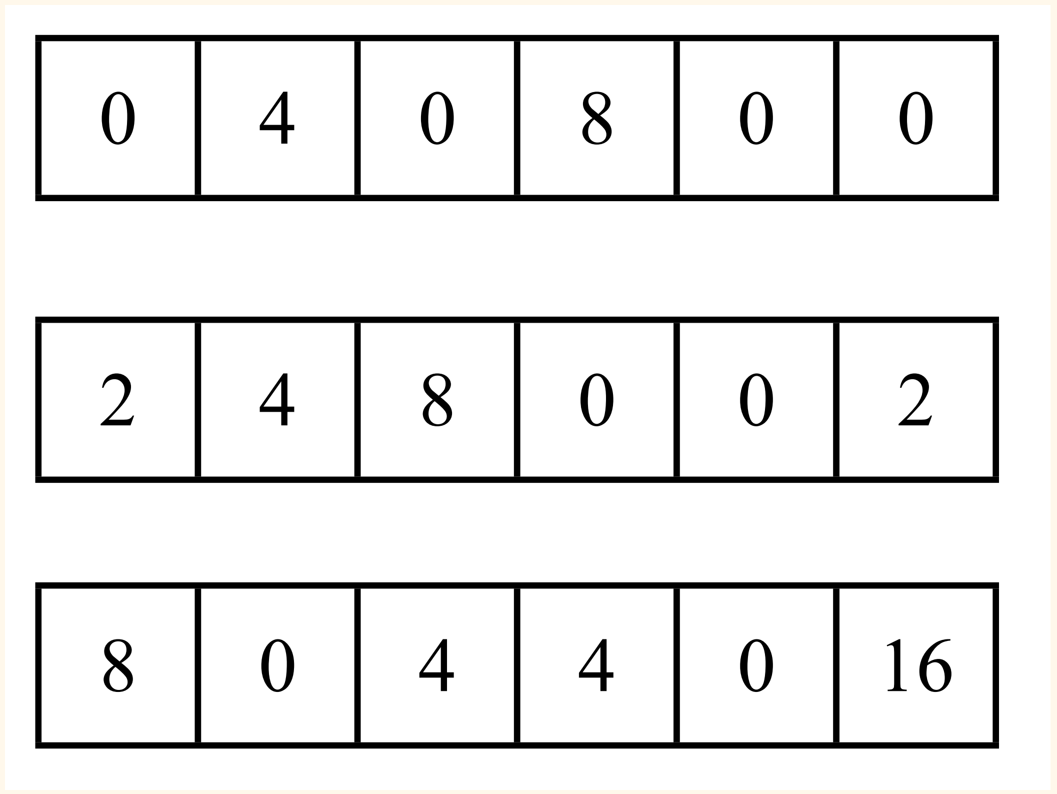
$$\vec{q}_m = \alpha \vec{q}_0 + \beta \frac{1}{|D_r|} \sum_{\vec{d}_j \in \underset{\substack{\uparrow \\ \text{small set of known-relevant docs}}}{D_r}} \vec{d}_j - \gamma \frac{1}{|D_{nr}|} \sum_{\vec{d}_j \in \underset{\substack{\uparrow \\ \text{small set of known-non-relevant docs}}}{D_{nr}}} \vec{d}_j$$

α, β, γ are weights for how much to rely on the various components.

In any event, positive feedback is almost always more useful, so $\beta > \gamma$.

The classical approach: Rocchio's algorithm

$$\vec{q}_m = \alpha \vec{q}_0 + \beta \frac{1}{|D_r|} \sum_{\vec{d}_j \in D_r} \vec{d}_j - \gamma \frac{1}{|D_{nr}|} \sum_{\vec{d}_j \in D_{nr}} \vec{d}_j$$



0	4	0	8	0	0	Original query vector
2	4	8	0	0	2	Known-relevant centroid
8	0	4	4	0	16	Known-non-relevant centroid

The classical approach: Rocchio's algorithm

$$\vec{q}_m = \alpha \vec{q}_0 + \beta \frac{1}{|D_r|} \sum_{\vec{d}_j \in D_r} \vec{d}_j - \gamma \frac{1}{|D_{nr}|} \sum_{\vec{d}_j \in D_{nr}} \vec{d}_j$$

0	4	0	8	0	0
2	4	8	0	0	2
8	0	4	4	0	16

$\alpha = 1$

$\beta = 0.5$

$\gamma = 0.25$

	0	4	0	8	0	0
+	1	2	4	0	0	1
-	2	0	1	1	0	4
	-1	6	3	7	0	-3
	0	6	3	7	0	0

Under Rocchio's formulation, negative weights get zeroed out.

Borrowed from: http://www.umiacs.umd.edu/~jimmylin/LBSC796-INFM718R-2006-Spring/syllabus.html

Another formulation: Probabilistic approaches

A simple naïve Bayesian model:

$$\begin{aligned}\hat{P}(x_t = 1|R = 1) &= |VR_t|/|VR| \\ \hat{P}(x_t = 1|R = 0) &= (df_t - \underset{\substack{\uparrow \\ \text{\# known-relevant documents with } t}}{|VR_t|})/(N - \underset{\substack{\uparrow \\ \text{\# known-relevant documents}}}{|VR|})\end{aligned}$$

These probabilities can be used to re-weight any other probabilistic retrieval model (e.g., the Lee & Croft paper).

Explicit relevance feedback imposes work on users…

Automated local analysis methods can help!

Pseudo-relevance feedback

Implicit feedback

Local clustering

Pseudo-relevance feedback is the simplest form of automated local analysis.

The idea: Do retrieval as normal;

Blindly take the top *k* returned documents, assume relevance;

Proceed with relevance feedback as normal.

Amazingly, this (kind of) works!

Problems: susceptible to topic drift, etc.

Implicit feedback: use click-through data, etc. to determine relevance.

(discussed previously, and again in a future lecture)

Local clustering:

Identify possible query expansion terms by doing associational clustering within the retrieved result set.

Can use local context to generate more useful expansions.

Can operate at the document level, or even at the passage level within a document.

Manning and Schütze make a very good point:

“Relevance feedback can improve both recall and precision. But, in practice, it has been shown to be most useful for increasing recall in situations where recall is important. This is partly because the technique expands the query, but it is also partly an effect of the use case: when they want high recall, users can be expected to take time to review results and to iterate on the search.”

How should we evaluate relevance feedback?

One idea: look at MAP before and after feedback, on the same set of documents.

What's wrong with this picture?

It's cheating! For the second query, we know *a priori* some documents to include!

How should we evaluate relevance feedback?

Second idea: Evaluate on the *residual collection* (the set of documents excluding those judged by the user)

What's wrong with this picture?

We get artificially degraded performance, since we exclude some of the most relevant documents.

If our goal is to compare two different RF approaches, this may not be an issue.

How should we evaluate relevance feedback?

Third idea: Use multiple sets of documents ("train"/"test"), examine post-RF query on both.

Fourth idea: Extrinsic evaluation!

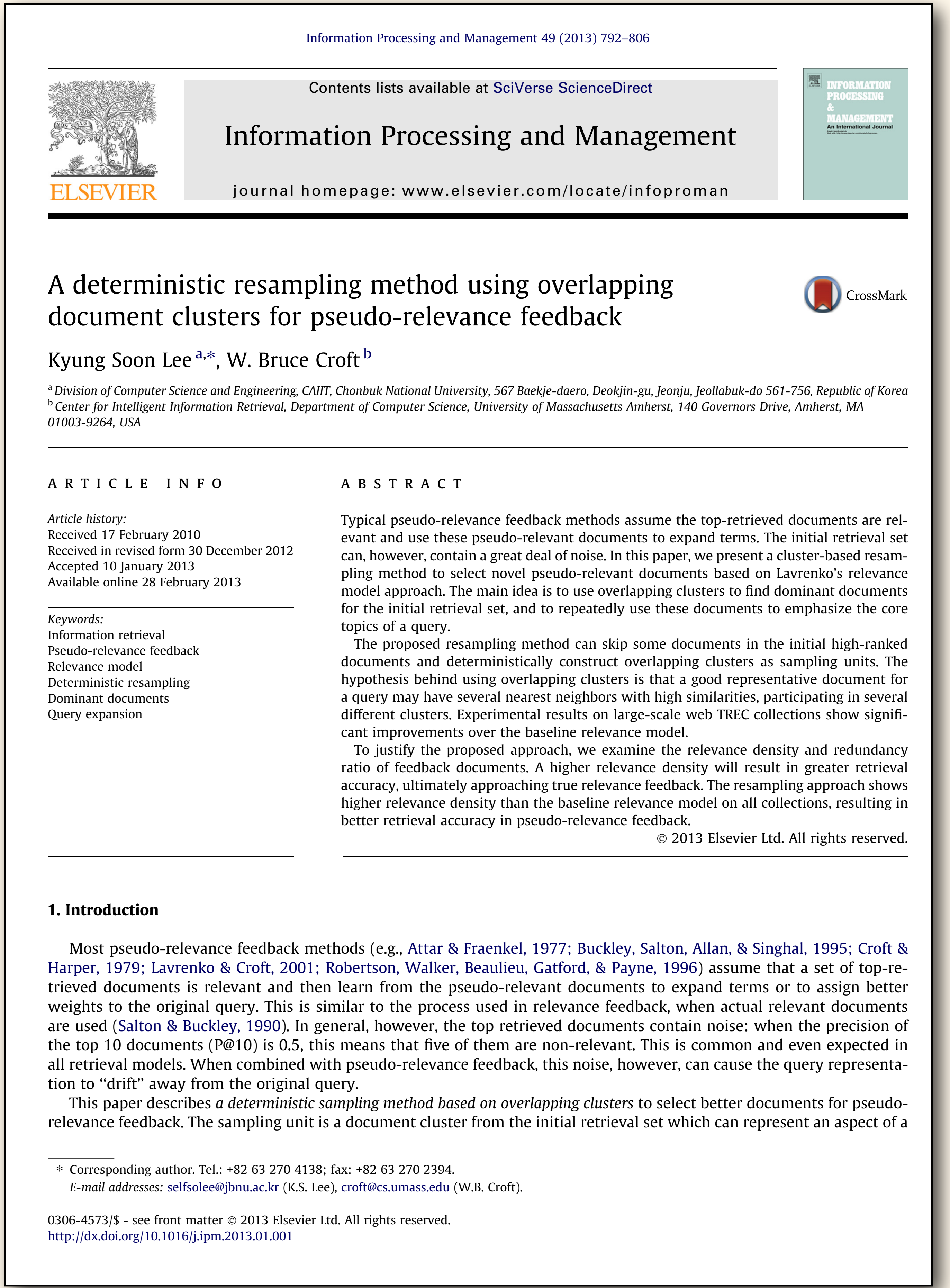
Does RF help the user complete a task more quickly? Do they identify more relevant documents using RF? Etc. etc.

Problems with Relevance Feedback:

RF does not help with:

- Mis-spellings
- Mismatch of searcher's vocabulary to collection's ("laptop" vs. "notebook")
- Documents not clustering "naturally"
- Relevance not being related to term distribution of documents

Also: users often don't like giving explicit feedback!



Information Processing and Management 49 (2013) 792–806

Contents lists available at SciVerse ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

A deterministic resampling method using overlapping document clusters for pseudo-relevance feedback

Kyung Soon Lee[a,*], W. Bruce Croft[b]

[a] Division of Computer Science and Engineering, CAIIT, Chonbuk National University, 567 Baekje-daero, Deokjin-gu, Jeonju, Jeollabuk-do 561-756, Republic of Korea
[b] Center for Intelligent Information Retrieval, Department of Computer Science, University of Massachusetts Amherst, 140 Governors Drive, Amherst, MA 01003-9264, USA

ARTICLE INFO

Article history:
Received 17 February 2010
Received in revised form 30 December 2012
Accepted 10 January 2013
Available online 28 February 2013

Keywords:
Information retrieval
Pseudo-relevance feedback
Relevance model
Deterministic resampling
Dominant documents
Query expansion

ABSTRACT

Typical pseudo-relevance feedback methods assume the top-retrieved documents are relevant and use these pseudo-relevant documents to expand terms. The initial retrieval set can, however, contain a great deal of noise. In this paper, we present a cluster-based resampling method to select novel pseudo-relevant documents based on Lavrenko's relevance model approach. The main idea is to use overlapping clusters to find dominant documents for the initial retrieval set, and to repeatedly use these documents to emphasize the core topics of a query.

The proposed resampling method can skip some documents in the initial high-ranked documents and deterministically construct overlapping clusters as sampling units. The hypothesis behind using overlapping clusters is that a good representative document for a query may have several nearest neighbors with high similarities, participating in several different clusters. Experimental results on large-scale web TREC collections show significant improvements over the baseline relevance model.

To justify the proposed approach, we examine the relevance density and redundancy ratio of feedback documents. A higher relevance density will result in greater retrieval accuracy, ultimately approaching true relevance feedback. The resampling approach shows higher relevance density than the baseline relevance model on all collections, resulting in better retrieval accuracy in pseudo-relevance feedback.

1. Introduction

Most pseudo-relevance feedback methods (e.g., Attar & Fraenkel, 1977; Buckley, Salton, Allan, & Singhal, 1995; Croft & Harper, 1979; Lavrenko & Croft, 2001; Robertson, Walker, Beaulieu, Gatford, & Payne, 1996) assume that a set of top-retrieved documents is relevant and then learn from the pseudo-relevant documents to expand terms or to assign better weights to the original query. This is similar to the process used in relevance feedback, when actual relevant documents are used (Salton & Buckley, 1990). In general, however, the top retrieved documents contain noise: when the precision of the top 10 documents (P@10) is 0.5, this means that five of them are non-relevant. This is common and even expected in all retrieval models. When combined with pseudo-relevance feedback, this noise, however, can cause the query representation to "drift" away from the original query.

This paper describes *a deterministic sampling method based on overlapping clusters* to select better documents for pseudo-relevance feedback. The sampling unit is a document cluster from the initial retrieval set which can represent an aspect of a

* Corresponding author. Tel.: +82 63 270 4138; fax: +82 63 270 2394.
E-mail addresses: selfsolee@jbnu.ac.kr (K.S. Lee), croft@cs.umass.edu (W.B. Croft).

0306-4573/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.ipm.2013.01.001

The basic idea:

Pseudo-relevance feedback (PRF) is easy to do...

... but obviously depends heavily on the relevance of the first several documents.

Lee & Croft's approach: be more clever about selecting which documents to use for PRF.

The main idea:

Cluster result documents according to similarity;

Identify “dominant” documents (those that appear in multiple clusters)...

... use those to select query expansion terms.

The initial set of results is produced using standard language model-based retrieval:

$$P(Q|D) = \prod_{i=1}^{m} P(q_i|D)$$

The initial set of results is produced using standard language model-based retrieval.

The results are *k*-means clustered, and then each cluster is sampled...

$$P(Q|Clu) = \prod_{i=1}^{m} P(q_i|Clu)$$

Collection	LM	Rerank	RM	Resampling	TrueRF
GOV2	0.3258	0.3406$^{\alpha}$	0.3581$^{\alpha\beta}$	**0.3806**$^{\alpha\beta\gamma}$	0.4315$^{\alpha\beta\gamma\delta}$
WT10g	0.1861	0.2044$^{\alpha}$	0.1966	**0.2352**$^{\alpha\beta\gamma}$	0.4030$^{\alpha\beta\gamma\delta}$
ROBUST	0.2920	0.3206$^{\alpha}$	**0.3591**$^{\alpha\beta}$	0.3515$^{\alpha\beta}$	0.5351$^{\alpha\beta\gamma\delta}$
AP	0.2077	0.2361$^{\alpha}$	0.2803$^{\alpha\beta}$	**0.2906**$^{\alpha\beta}$	0.4253$^{\alpha\beta\gamma\delta}$
WSJ	0.3258	0.3611$^{\alpha}$	0.3967$^{\alpha\beta}$	**0.4033**$^{\alpha\beta}$	0.5306$^{\alpha\beta\gamma\delta}$