Introduction to

Information Retrieval

Hinrich Schütze and Christina Lioma

Lecture 9: Relevance Feedback & Query Expansion

Take-away today

- Interactive relevance feedback: improve initial retrieval results by telling the IR system which docs are relevant / nonrelevant
- Best known relevance feedback method: Rocchio feedback
- Query expansion: improve retrieval results by adding synonyms / related terms to the query
  - Sources for related terms: Manual thesauri, automatic thesauri, query logs

Overview

1. Motivation
2. Relevance feedback: Basics
3. Relevance feedback: Details
4. Query expansion

Outline

1. Motivation
2. Relevance feedback: Basics
3. Relevance feedback: Details
4. Query expansion

How can we improve recall in search?

- Main topic today: two ways of improving recall: relevance feedback and query expansion
- As an example consider query *q*: [aircraft] . . .
- . . . and document *d* containing "plane", but not containing "aircraft"
- A simple IR system will not return *d* for *q*.
- Even if *d* is the most relevant document for *q*!
- We want to change this:
- Return relevant documents even if there is no term match with the (original) query

Recall

- Loose definition of recall in this lecture: “increasing the number of relevant documents returned to user”
- This may actually decrease recall on some measures, e.g., when expanding “jaguar” with “panthera”
  - . . .which eliminates some relevant documents, but increases relevant documents returned on top pages

Options for improving recall

- Local: Do a “local”, on-demand analysis for a user query
  - Main local method: relevance feedback
  - Part 1
- Global: Do a global analysis once (e.g., of collection) to produce thesaurus
  - Use thesaurus for query expansion
  - Part 2

Google examples for query expansion

- One that works well
  - *~flights -flight*
- One that doesn't work so well
  - *~hospitals -hospital*

Outline

1. Motivation
2. Relevance feedback: Basics
3. Relevance feedback: Details
4. Query expansion

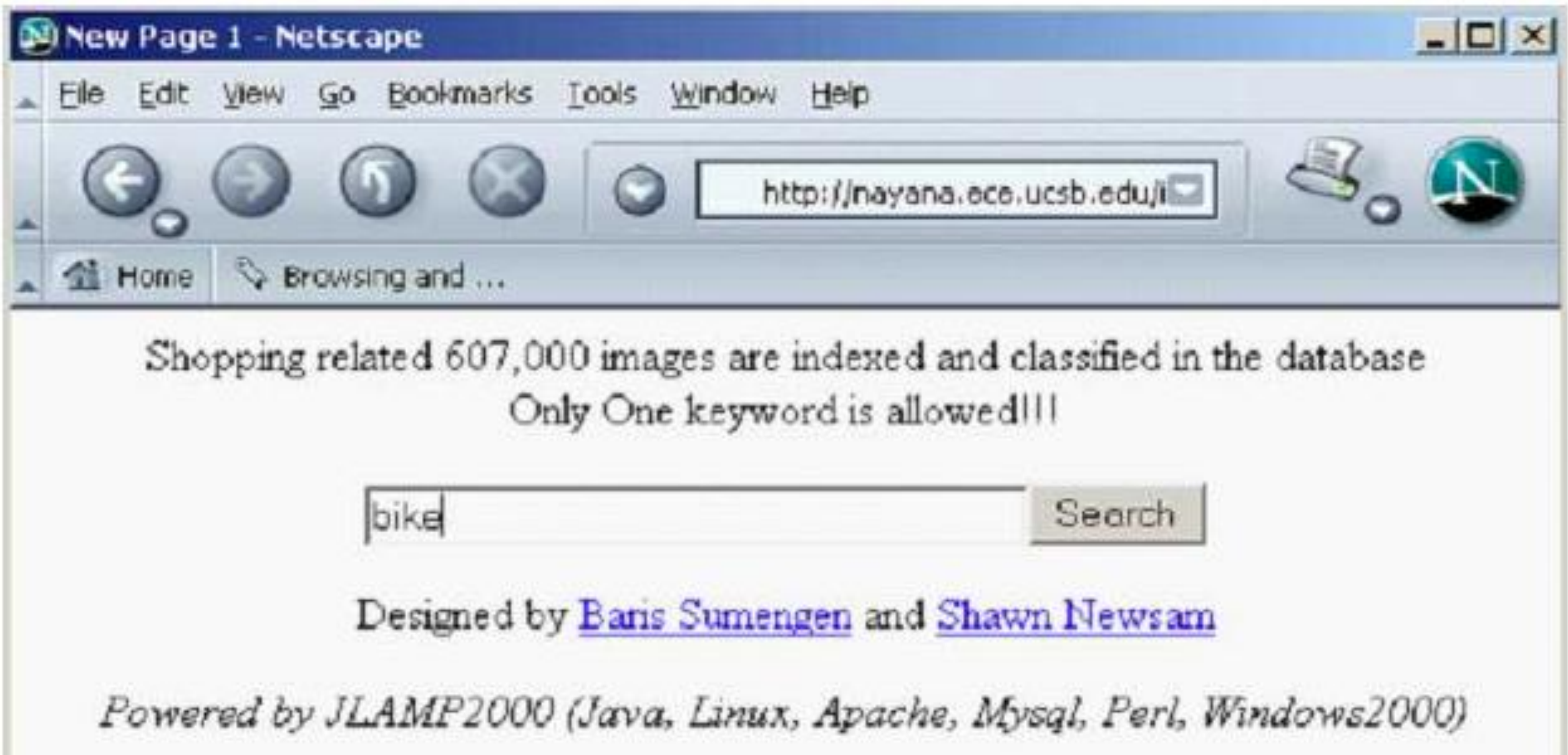
Relevance feedback: Basic idea

- The user issues a (short, simple) query.
- The search engine returns a set of documents.
- User marks some docs as relevant, some as nonrelevant.
- Search engine computes a new representation of the information need. Hope: better than the initial query.
- Search engine runs new query and returns new results.
- New results have (hopefully) better recall.

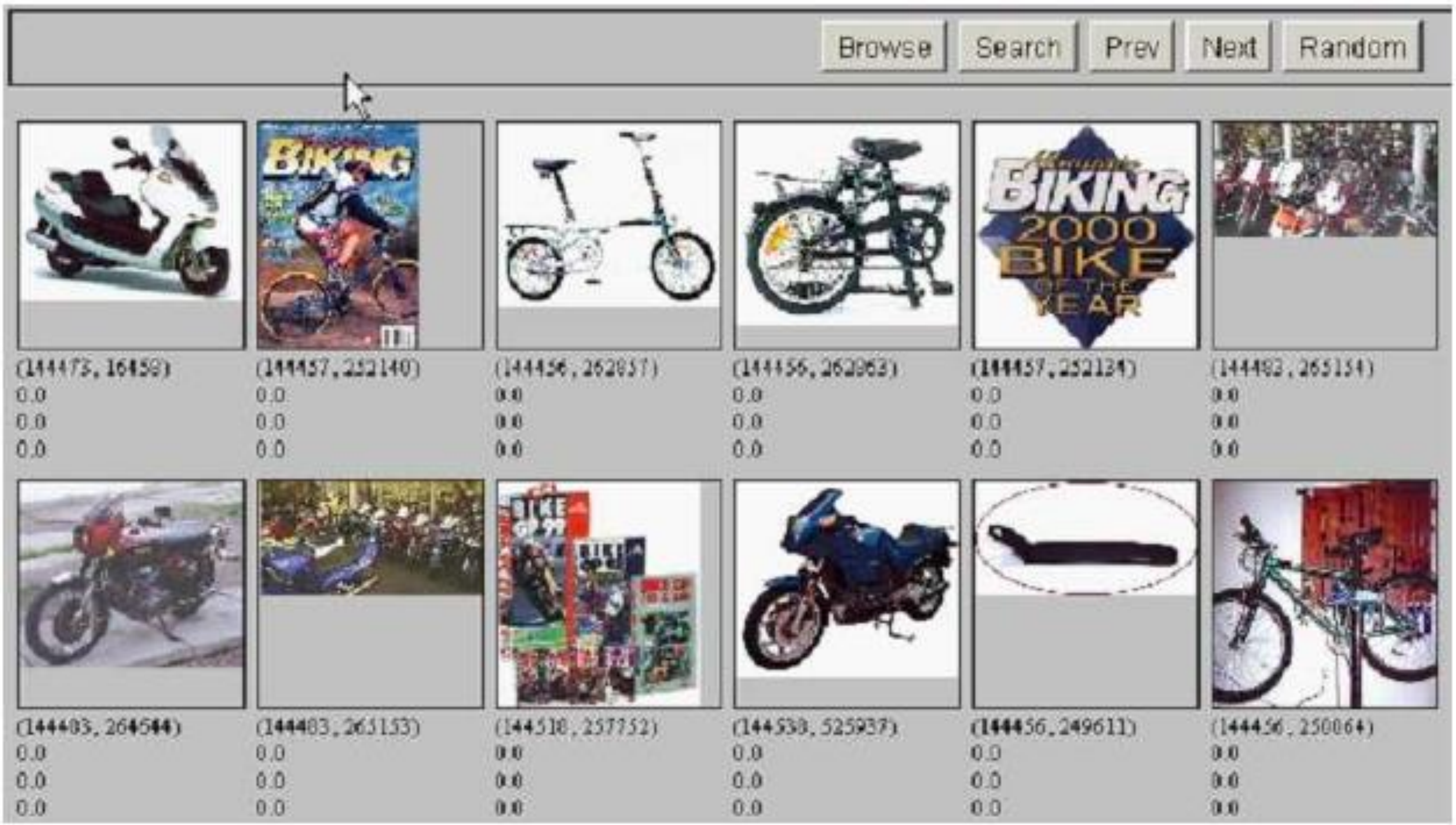
Relevance feedback

- We can iterate this: several rounds of relevance feedback.
- We will use the term ad hoc retrieval to refer to regular retrieval without relevance feedback.
- We will now look at three different examples of relevance feedback that highlight different aspects of the process.

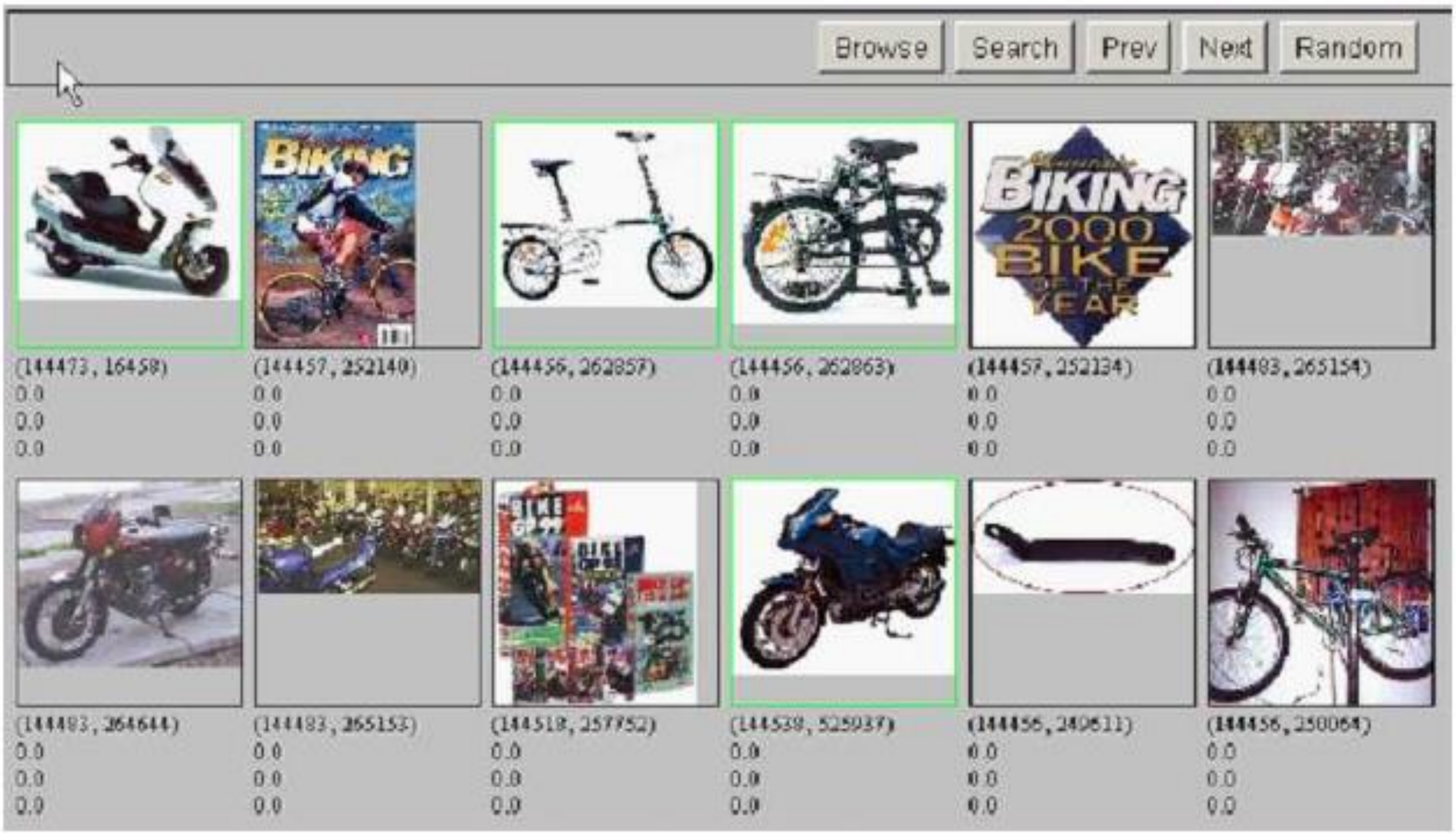
Relevance feedback: Example 1



Results for initial query

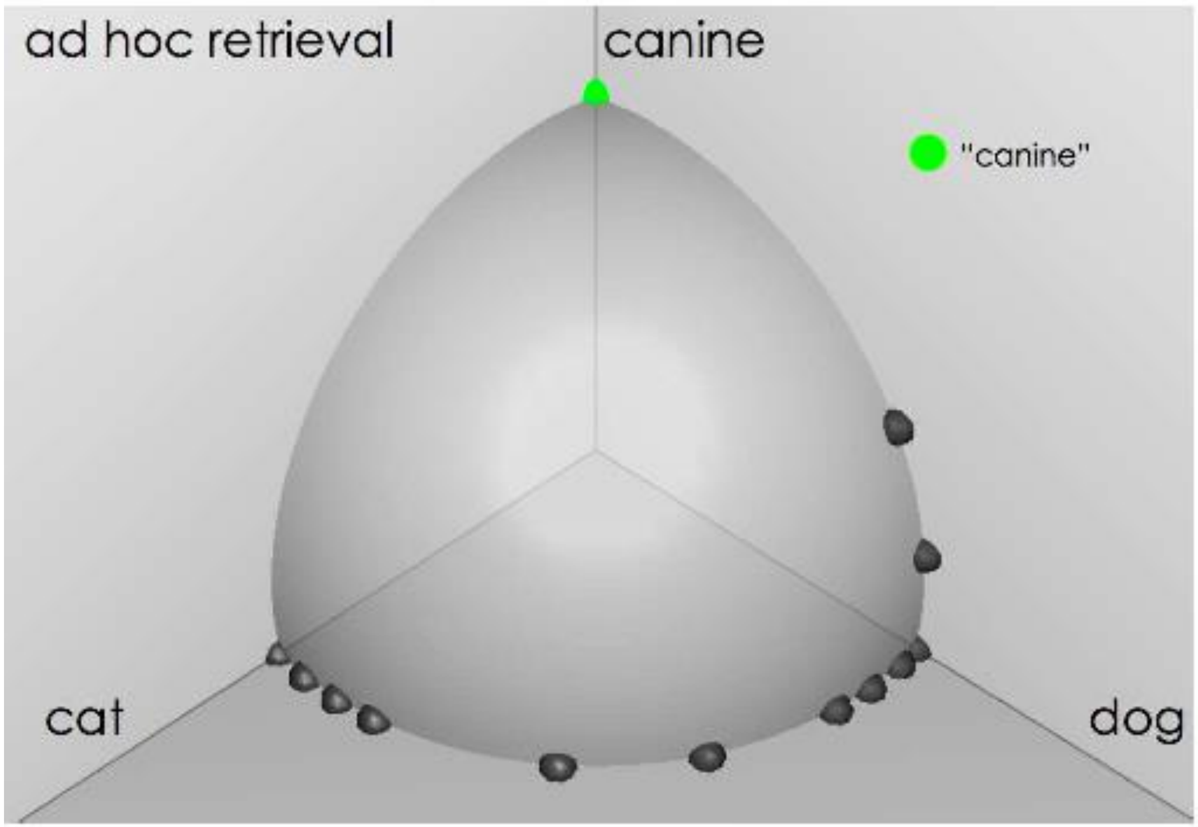


User feedback: Select what is relevant



Results after relevance feedback

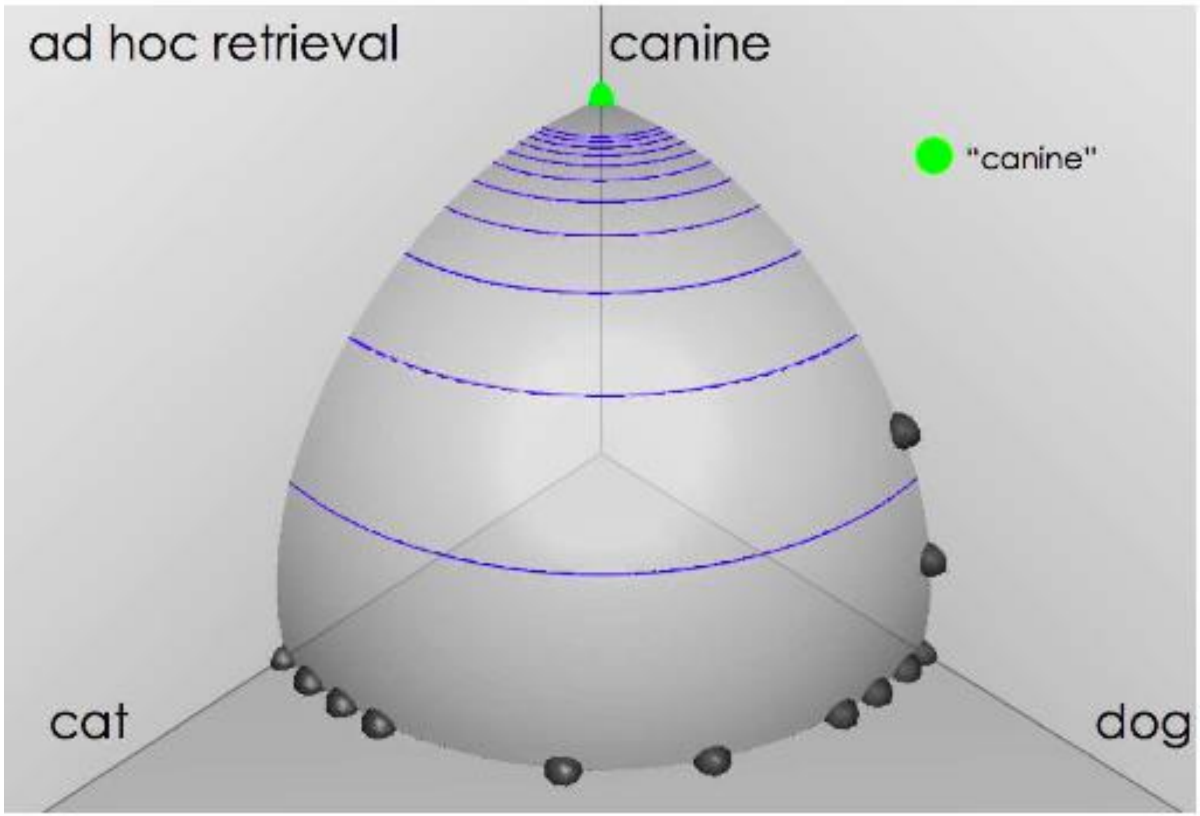
Vector space example: query "canine" (1)



Source:

Fernando Díaz

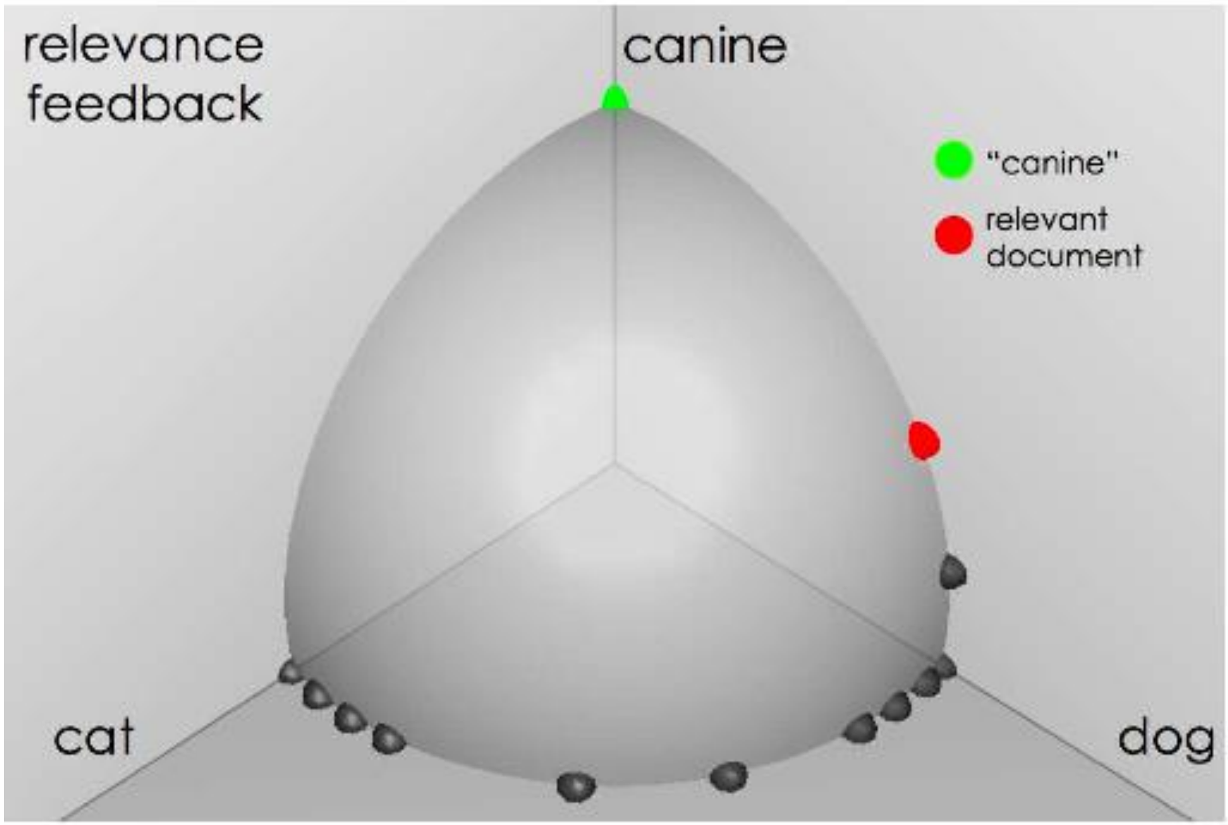
Similarity of docs to query "canine"



Source:

Fernando Díaz

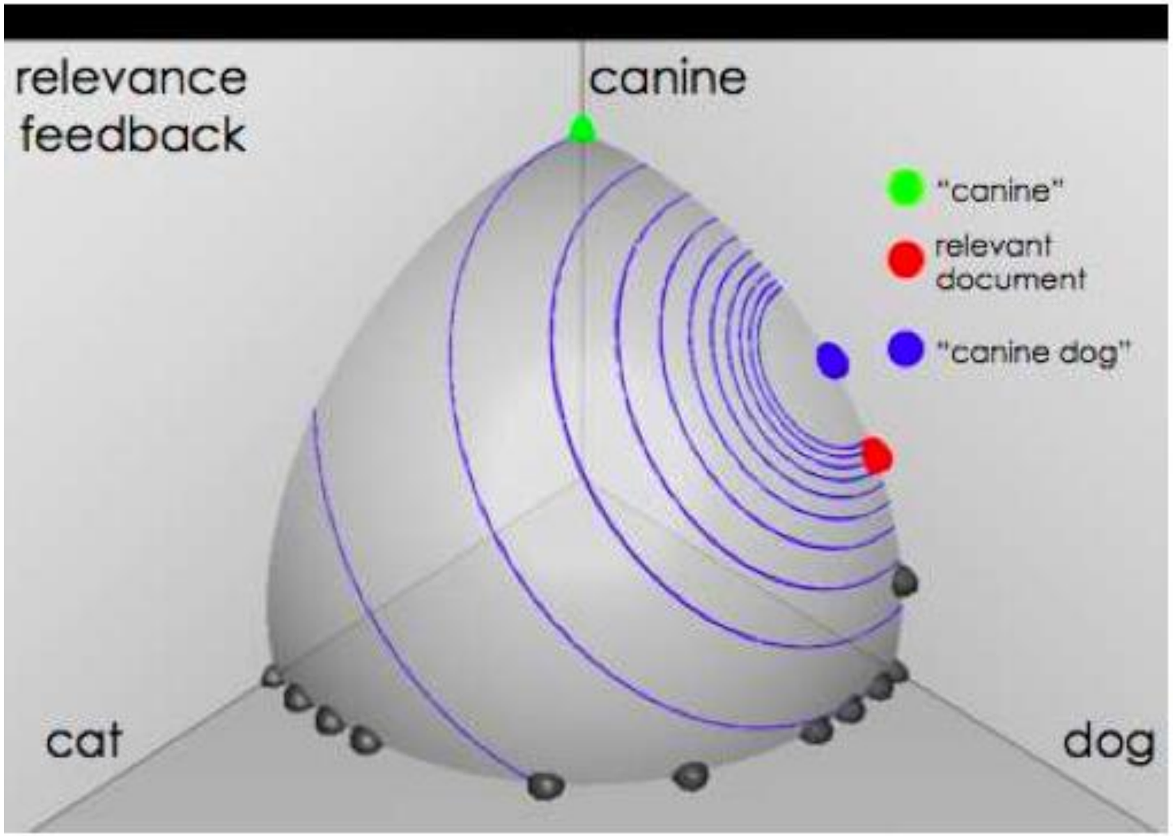
User feedback: Select relevant documents



Source:

Fernando Díaz

Results after relevance feedback



Source:

Fernando Díaz

Example 3: A real (non-image) example

Initial query:

[new space satellite applications] Results for initial query: (*r* = rank)

	r		
+	1	0.539	NASA Hasn't Scrapped Imaging Spectrometer
+	2	0.533	NASA Scratches Environment Gear From Satellite Plan
	3	0.528	Science Panel Backs NASA Satellite Plan, But Urges Launches of Smaller Probes
	4	0.526	A NASA Satellite Project Accomplishes Incredible Feat: Staying Within Budget
	5	0.525	Scientist Who Exposed Global Warming Proposes Satellites for Climate Research
	6	0.524	Report Provides Support for the Critics Of Using Big Satellites to Study Climate
	7	0.516	Arianespace Receives Satellite Launch Pact From Telesat Canada
+	8	0.509	Telecommunications Tale of Two Companies

User then marks relevant documents with "+".

Expanded query after relevance feedback

2.074	new	15.106	space
30.816	satellite	5.660	application
5.991	nasa	5.196	eos
4.196	launch	3.972	aster
3.516	instrument	3.446	arianespace
3.004	bundespost	2.806	ss
2.790	rocket	2.053	scientist
2.003	broadcast	1.172	earth
0.836	oil	0.646	measure

Compare to original

query: [new space satellite applications]

Results for expanded query

	r		
*	1	0.513	NASA Scratches Environment Gear From Satellite Plan
*	2	0.500	NASA Hasn't Scrapped Imaging Spectrometer
	3	0.493	When the Pentagon Launches a Secret Satellite, Space Sleuths Do Some Spy Work of Their Own
	4	0.493	NASA Uses 'Warm' Superconductors For Fast Circuit
*	5	0.492	Telecommunications Tale of Two Companies
	6	0.491	Soviets May Adapt Parts of SS-20 Missile For Commercial Use
	7	0.490	Gaping Gap: Pentagon Lags in Race To Match the Soviets In Rocket Launchers
	8	0.490	Rescue of Satellite By Space Agency To Cost $90 Million

Outline

1. Motivation
2. Relevance feedback: Basics
3. Relevance feedback: Details
4. Query expansion

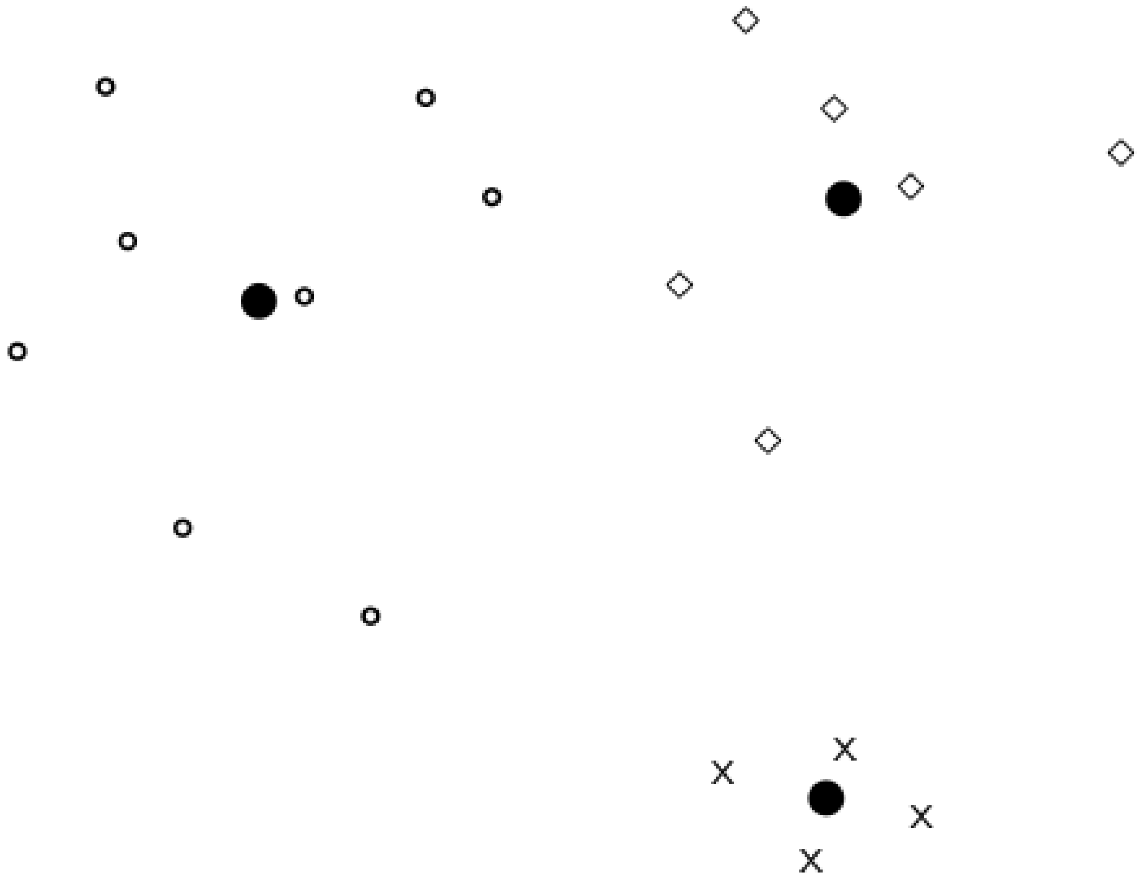
Key concept for relevance feedback: Centroid

- The centroid is the center of mass of a set of points.
- Recall that we represent documents as points in a high-dimensional space.
- Thus: we can compute centroids of documents.
- Definition:

$$\vec{\mu}(D) = \frac{1}{|D|} \sum_{d \in D} \vec{v}(d)$$

where D is a set of documents and $\vec{v}(d) = \vec{d}$ is the vector we use to represent document *d*.

Centroid: Example



Rocchio' algorithm

- The Rocchio' algorithm implements relevance feedback in the vector space model.
- Rocchio' chooses the query $\vec{q}_{opt}$ that maximizes

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

D_r : set of relevant docs; D_{nr} : set of nonrelevant docs

- Intent: ~qopt is the vector that separates relevant and nonrelevant docs maximally.
- Making some additional assumptions, we can rewrite $\vec{q}_{opt}$ as:

$$\vec{q}_{opt} = \mu(D_r) + [\mu(D_r) - \mu(D_{nr})]$$

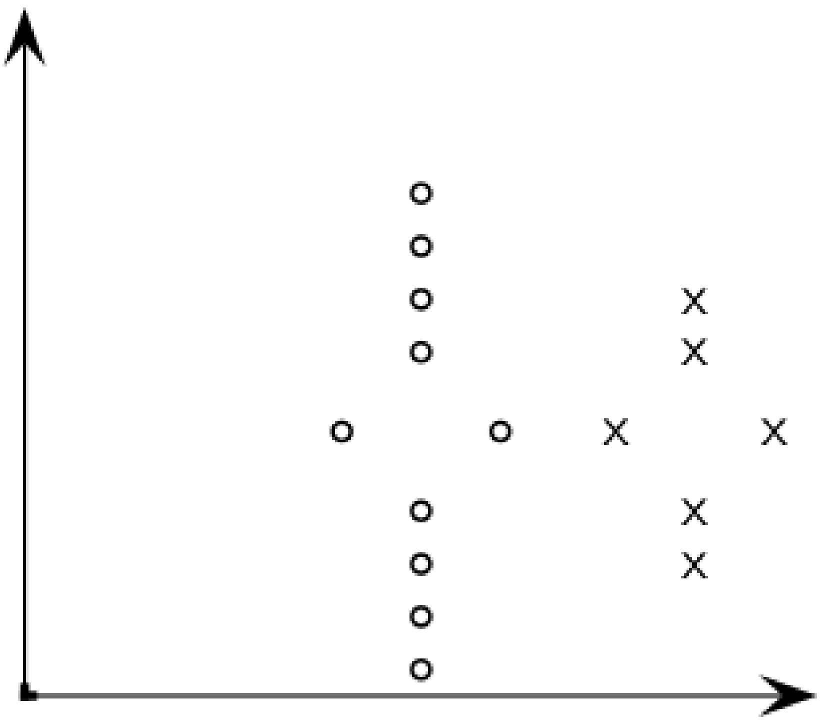
Rocchio' algorithm

- The optimal query vector is:

$$\begin{aligned}\vec{q}_{opt} &= \mu(D_r) + [\mu(D_r) - \mu(D_{nr})] \\ &= \frac{1}{|D_r|}\sum_{\vec{d}_j \in D_r} \vec{d}_j + [\frac{1}{|D_r|}\sum_{\vec{d}_j \in D_r} \vec{d}_j - \frac{1}{|D_{nr}|}\sum_{\vec{d}_j \in D_{nr}} \vec{d}_j]\end{aligned}$$

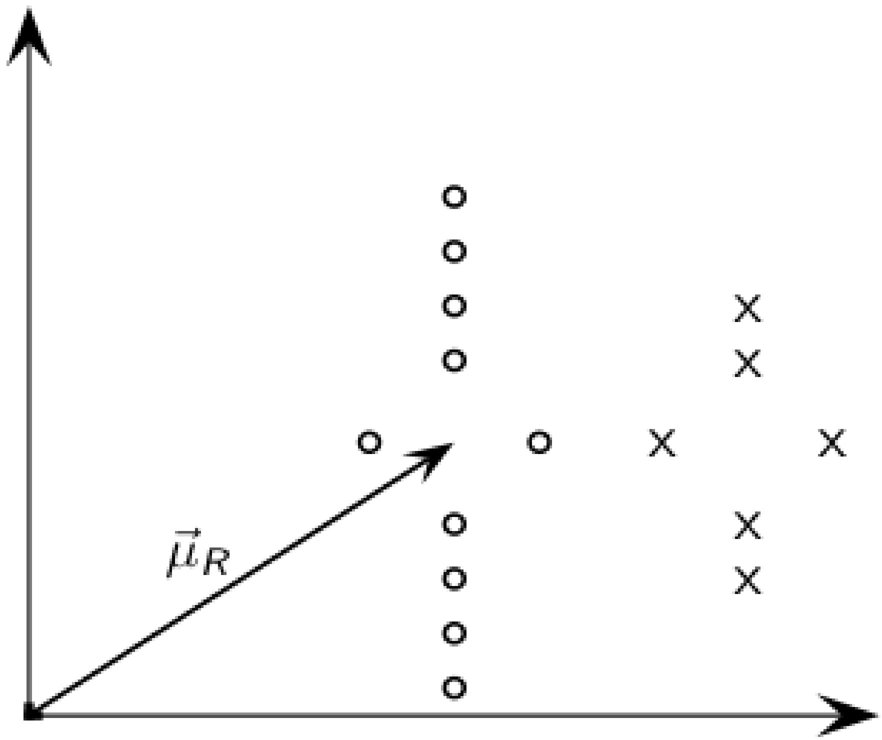
- We move the centroid of the relevant documents by the difference between the two centroids.

Exercise: Compute Rocchio' vector



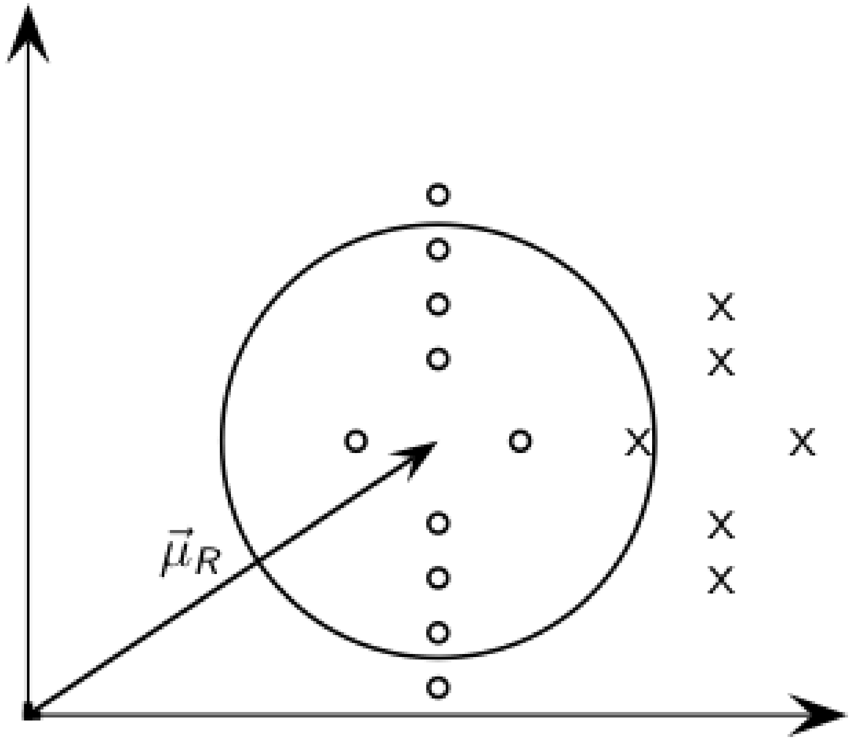
circles: relevant documents, Xs: nonrelevant documents

Rocchio' illustrated



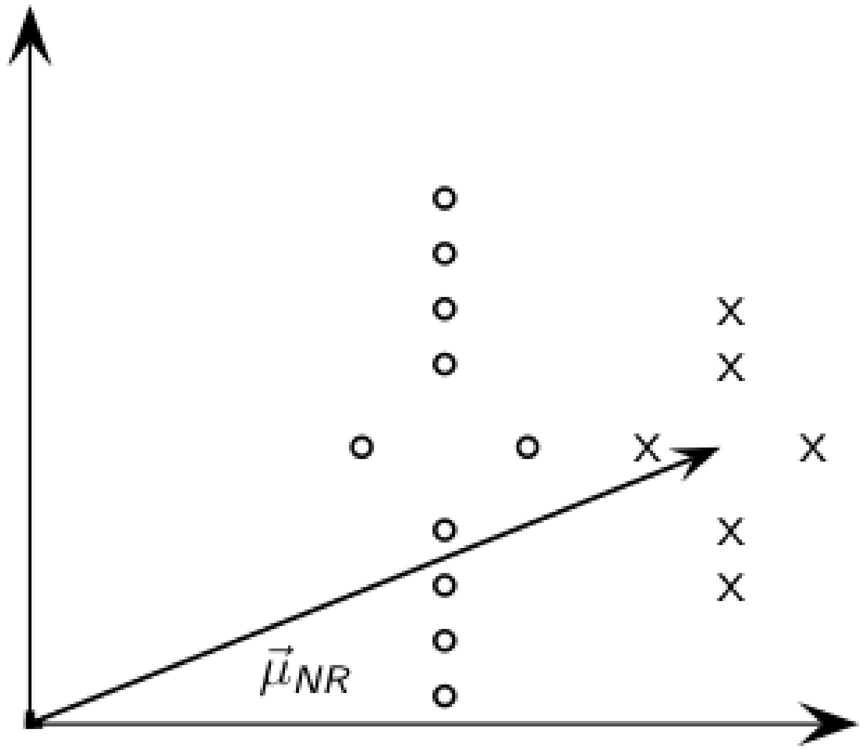
$\vec{\mu}_R$: centroid of relevant documents

Rocchio' illustrated



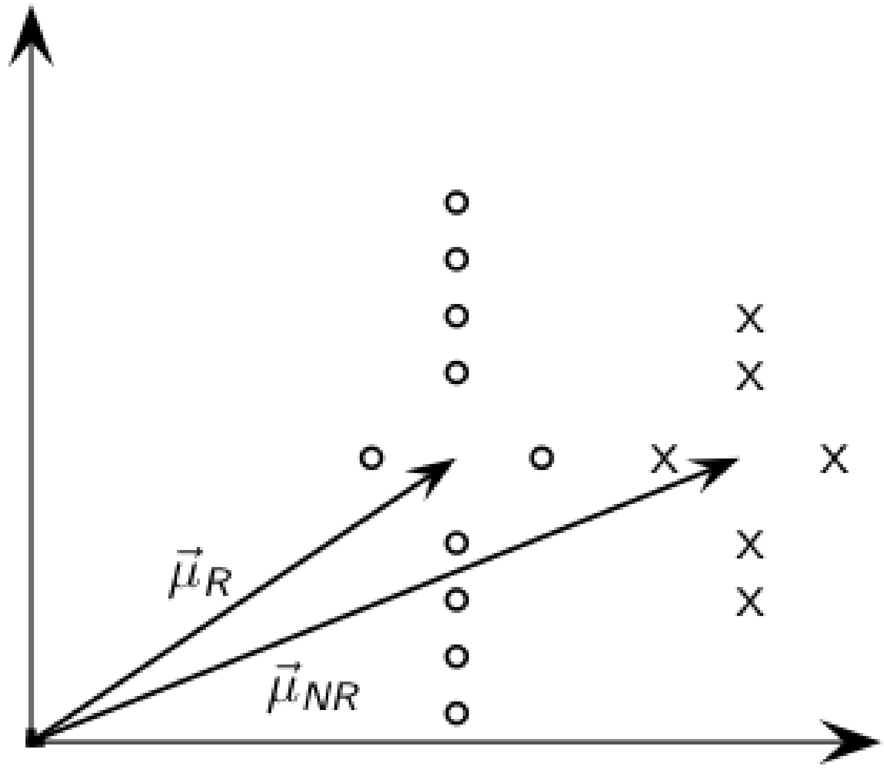
$\vec{\mu}_R$ does not separate relevant / nonrelevant.

Rocchio' illustrated

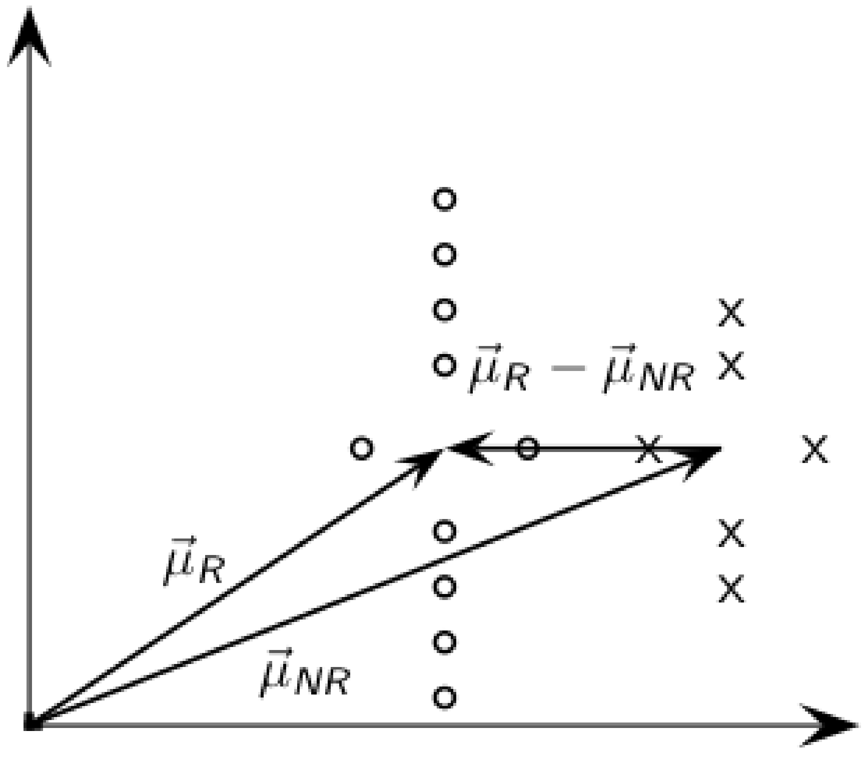


$\vec{\mu}_{NR}$: centroid of nonrelevant documents.

Rocchio' illustrated

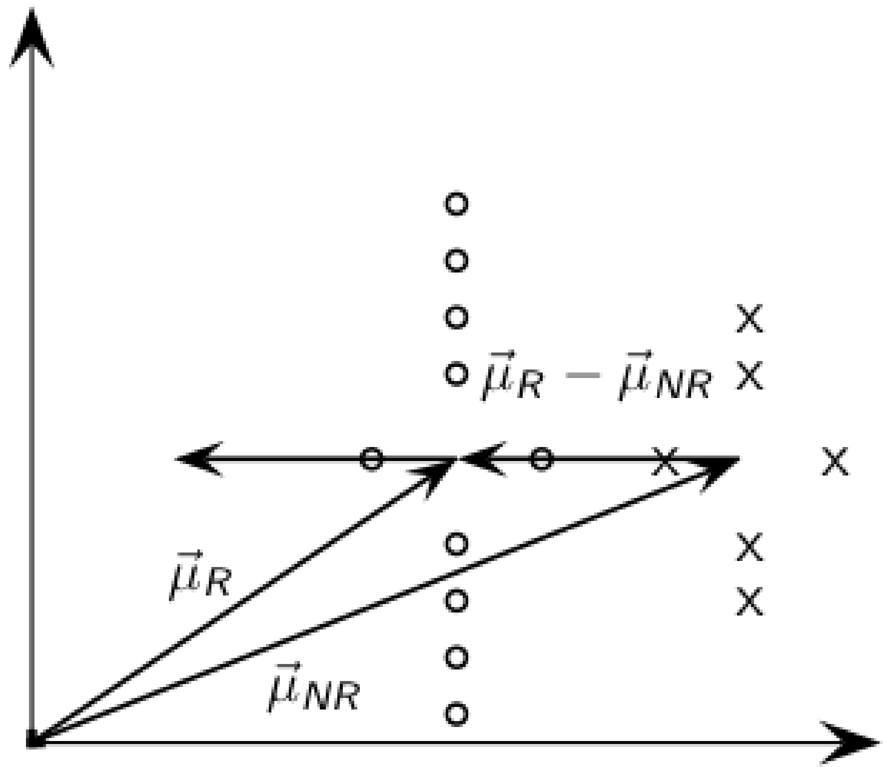


Rocchio' illustrated



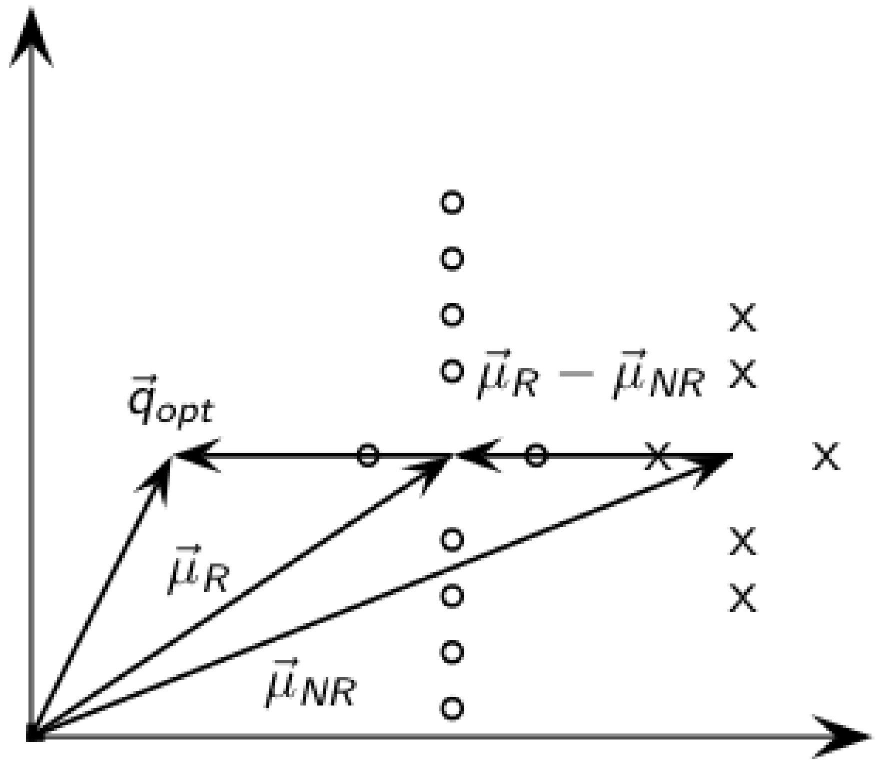
$\vec{\mu}_R$ - $\vec{\mu}_{NR}$: difference vector

Rocchio' illustrated



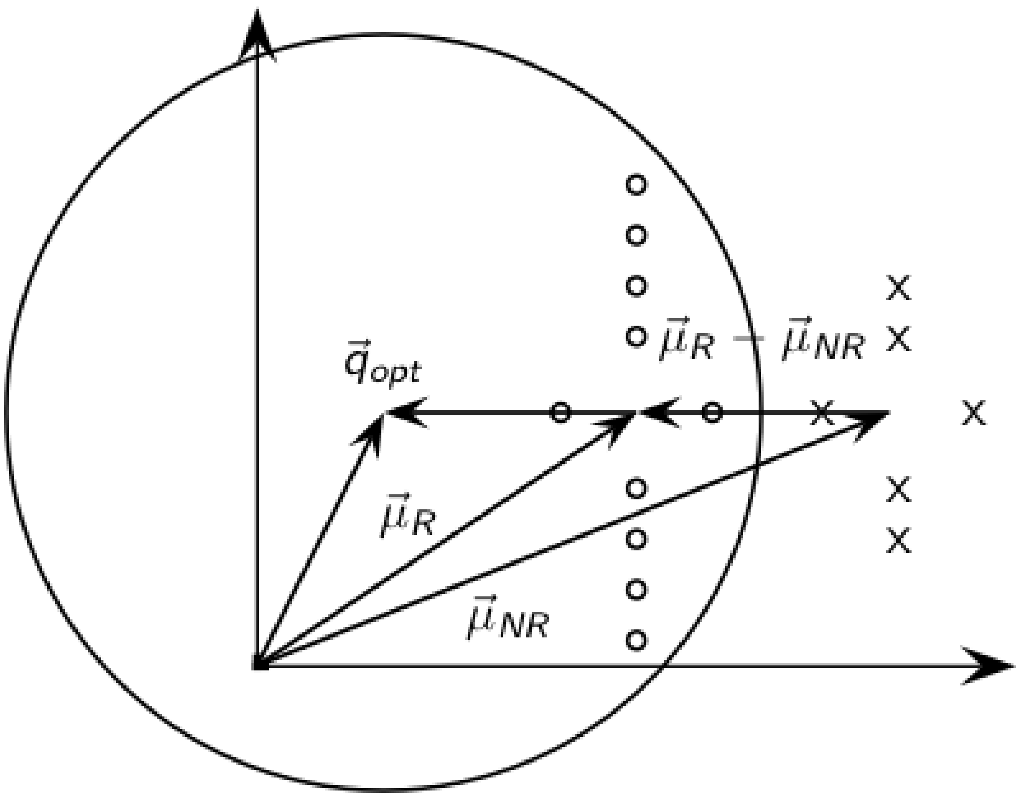
Add difference vector to $\vec{\mu}_R$ …

Rocchio' illustrated



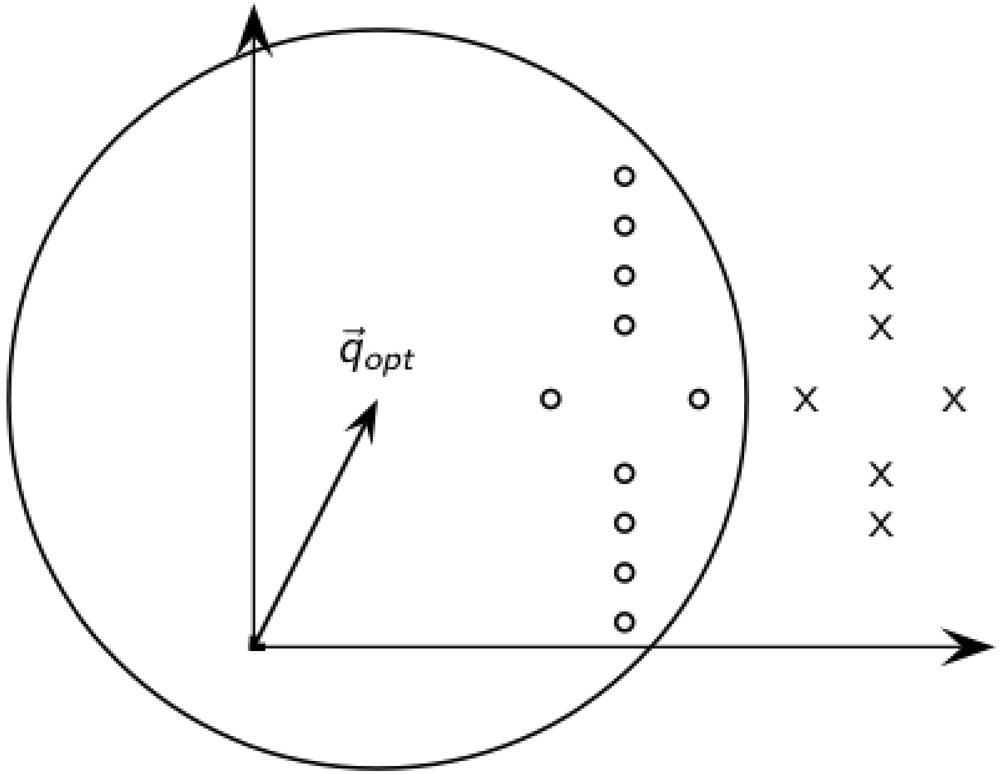
… to get $\vec{q}_{opt}$

Rocchio' illustrated



$\vec{q}_{opt}$ separates relevant / nonrelevant perfectly.

Rocchio' illustrated



$\vec{q}_{opt}$ separates relevant / nonrelevant perfectly.

Terminology

- We use the name Rocchio’ for the theoretically better motivated original version of Rocchio.
- The implementation that is actually used in most cases is the SMART implementation – we use the name Rocchio (without prime) for that.

Rocchio 1971 algorithm (SMART)

Used in practice:

$$\begin{aligned}\vec{q}_m &= \alpha\vec{q}_0 + \beta\mu(D_r) - \gamma\mu(D_{nr}) \\ &= \alpha\vec{q}_0 + \beta\frac{1}{|D_r|}\sum_{\vec{d}_j\in D_r}\vec{d}_j - \gamma\frac{1}{|D_{nr}|}\sum_{\vec{d}_j\in D_{nr}}\vec{d}_j\end{aligned}$$

q_m: modified query vector; q_0: original query vector; D_r and D_{nr} : sets of known relevant and nonrelevant documents respectively; α, β, and γ: weights

- New query moves towards relevant documents and away from nonrelevant documents.
- Tradeoff α vs. β/γ: If we have a lot of judged documents, we want a higher β/γ.
- Set negative term weights to 0.
- "Negative weight" for a term doesn't make sense in the vector space model.

Positive vs. negative relevance feedback

- Positive feedback is more valuable than negative feedback.
- For example, set $\beta = 0.75$, $\gamma = 0.25$ to give higher weight to positive feedback.
- Many systems only allow positive feedback.

Relevance feedback: Assumptions

- When can relevance feedback enhance recall?
- Assumption A1: The user knows the terms in the collection well enough for an initial query.
- Assumption A2: Relevant documents contain similar terms (so I can “hop” from one relevant document to a different one when giving relevance feedback).

Violation of A1

- Assumption A1: The user knows the terms in the collection well enough for an initial query.
- Violation: Mismatch of searcher's vocabulary and collection vocabulary
- Example: cosmonaut / astronaut

Violation of A2

- Assumption A2: Relevant documents are similar.
- Example for violation: [contradictory government policies]
- Several unrelated “prototypes”
  - Subsidies for tobacco farmers vs. anti-smoking campaigns
  - Aid for developing countries vs. high tariffs on imports from developing countries
- Relevance feedback on tobacco docs will not help with finding docs on developing countries.

Relevance feedback: Evaluation

- Pick one of the evaluation measures from last lecture, e.g., precision in top 10: *P*@10
- Compute *P*@10 for original query q_0
- Compute *P*@10 for modified relevance feedback query q1
- In most cases: q_1 is spectacularly better than q_0!
- Is this a fair evaluation?

Relevance feedback: Evaluation

- Fair evaluation must be on “residual” collection: docs not yet judged by user.
- Studies have shown that relevance feedback is successful when evaluated this way.
- Empirically, one round of relevance feedback is often very useful. Two rounds are marginally useful.

Evaluation: Caveat

- True evaluation of usefulness must compare to other methods taking the same amount of time.
- Alternative to relevance feedback: User revises and resubmits query.
- Users may prefer revision/resubmission to having to judge relevance of documents.
- There is no clear evidence that relevance feedback is the “best use” of the user’s time.

Exercise

- Do search engines use relevance feedback?
- Why?

Relevance feedback: Problems

- Relevance feedback is expensive.
  - Relevance feedback creates long modified queries.
  - Long queries are expensive to process.
- Users are reluctant to provide explicit feedback.
- It’s often hard to understand why a particular document was retrieved after applying relevance feedback.
- The search engine Excite had full relevance feedback at one point, but abandoned it later.

Pseudo-relevance feedback

- Pseudo-relevance feedback automates the “manual” part of true relevance feedback.
- Pseudo-relevance algorithm:
  - Retrieve a ranked list of hits for the user’s query
  - Assume that the top *k* documents are relevant.
  - Do relevance feedback (e.g., Rocchio)
- Works very well on average
- But can go horribly wrong for some queries.
- Several iterations can cause *query drift*.

Pseudo-relevance feedback at TREC4

- Cornell SMART system
- Results show number of relevant documents out of top 100 for 50 queries (so total number of documents is 5000):

method	number of relevant documents
lnc.ltc	3210
lnc.ltc-PsRF	3634
Lnu.ltu	3709
Lnu.ltu-PsRF	4350

- Results contrast two length normalization schemes (L vs. l) and pseudo-relevance feedback (PsRF).
- The pseudo-relevance feedback method used added only 20 terms to the query. (Rocchio will add many more.)
- This demonstrates that pseudo-relevance feedback is effective on average.

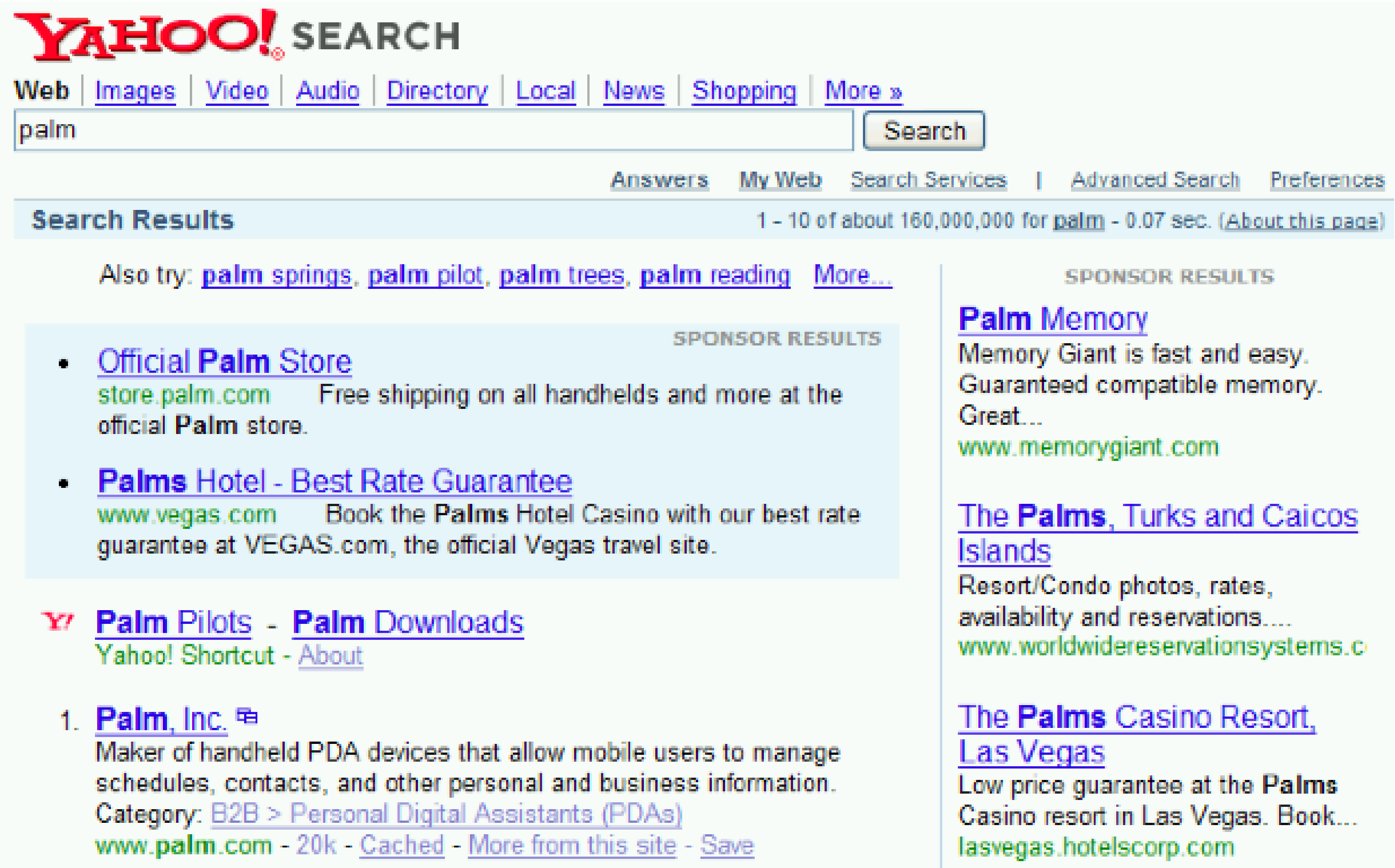
Outline

1. Motivation
2. Relevance feedback: Basics
3. Relevance feedback: Details
4. Query expansion

Query expansion

- Query expansion is another method for increasing recall.
- We use “global query expansion” to refer to “global methods for query reformulation”.
- In global query expansion, the query is modified based on some global resource, i.e. a resource that is not query-dependent.
- Main information we use: (near-)synonymy
- A publication or database that collects (near-)synonyms is called a thesaurus.
- We will look at two types of thesauri: manually created and automatically created.

Query expansion: Example



Types of user feedback

- User gives feedback on documents.
  - More common in relevance feedback
- User gives feedback on words or phrases.
  - More common in query expansion

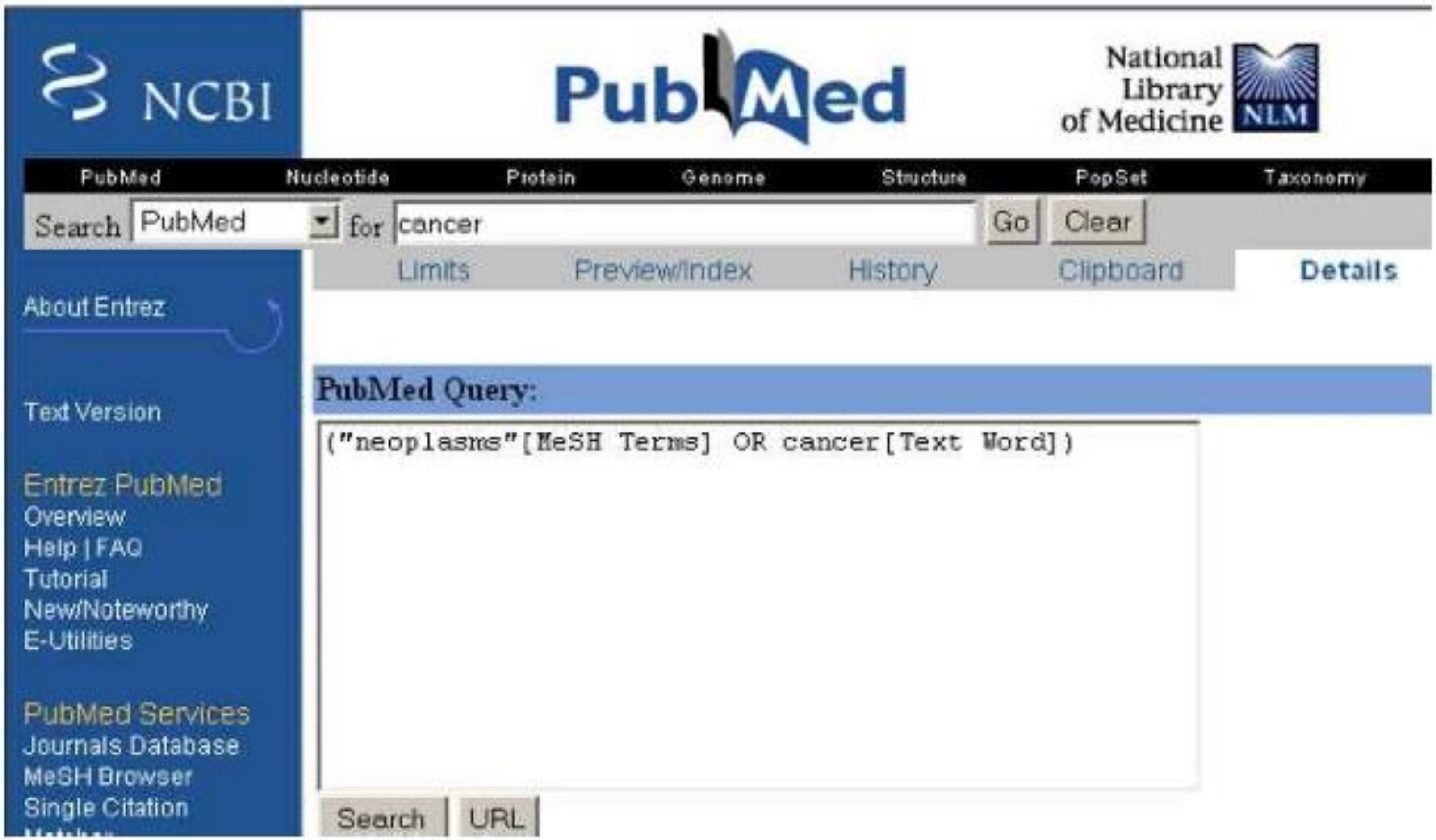
Types of query expansion

- Manual thesaurus (maintained by editors, e.g., PubMed)
- Automatically derived thesaurus (e.g., based on co-occurrence statistics)
- Query-equivalence based on query log mining (common on the web as in the “palm” example)

Thesaurus-based query expansion

- For each term t in the query, expand the query with words the thesaurus lists as semantically related with *t*.
- Example from earlier: HOSPITAL → MEDICAL
- Generally increases recall
- May significantly decrease precision, particularly with ambiguous terms
  - INTEREST RATE → INTEREST RATE FASCINATE
- Widely used in specialized search engines for science and engineering
- It's very expensive to create a manual thesaurus and to maintain it over time.
- A manual thesaurus has an effect roughly equivalent to annotation with a controlled vocabulary.

Example for manual thesaurus: PubMed



Automatic thesaurus generation

- Attempt to generate a thesaurus automatically by analyzing the distribution of words in documents
- Fundamental notion: similarity between two words
- Definition 1: Two words are similar if they co-occur with similar words.
  - “car” ≈ “motorcycle” because both occur with “road”, “gas” and “license”, so they must be similar.
- Definition 2: Two words are similar if they occur in a given grammatical relation with the same words.
  - You can harvest, peel, eat, prepare, etc. apples and pears, so apples and pears must be similar.
- Co-occurrence is more robust, grammatical relations are more accurate.

Co-occurence-based thesaurus: Examples

Word	Nearest neighbors
absolutely	absurd whatsoever totally exactly nothing
bottomed	dip copper drops topped slide trimmed
captivating	shimmer stunningly superbly plucky witty
doghouse	dog porch crawling beside downstairs
makeup	repellent lotion glossy sunscreen skin gel
mediating	reconciliation negotiate case conciliation
keeping	hoping bring wiping could some would
lithographs	drawings Picasso Dali sculptures Gauguin
pathogens	toxins bacteria organisms bacterial parasite
senses	grasp psyche truly clumsy naive innate

WordSpace demo on web

Query expansion at search engines

- Main source of query expansion at search engines: query logs
- Example 1: After issuing the query [herbs], users frequently search for [herbal remedies].
  - → "herbal remedies" is potential expansion of "herb".
- Example 2: Users searching for [flower pix] frequently click on the URL photobucket.com/flower. Users searching for [flower clipart] frequently click on the same URL.
  - → "flower clipart" and "flower pix" are potential expansions of each other.

Take-away today

- Interactive relevance feedback: improve initial retrieval results by telling the IR system which docs are relevant / nonrelevant
- Best known relevance feedback method: Rocchio feedback
- Query expansion: improve retrieval results by adding synonyms / related terms to the query
  - Sources for related terms: Manual thesauri, automatic thesauri, query logs

Resources

- Chapter 9 of IIR
- Resources at http://ifnlp.org/ir
  - Salton and Buckley 1990 (original relevance feedback paper)
  - Spink, Jansen, Ozmultu 2000: Relevance feedback at Excite
  - Schütze 1998: Automatic word sense discrimination (describes a simple method for automatic thesuarus generation)