Multimedia Information Retrieval and Technology

Lecture 10 Kappa Measure and Relevance Feedback

By : Fangyu wu

Room: SD555

Xi'an Jiaotong-Liverpool University

西交利物浦大学

Recap： Measuring relevance

To measure IR effectiveness, we need a test collection consisting of **Three elements**:

1. A benchmark document collection

   Must be representative

2. A benchmark suite of information needs, expressible as queries

   Again, representative

3. An assessment of either Relevant or Nonrelevant judgments for each query-document pair.

   How?

Kappa Measure

Measure how much agreement/disagreement between judges there is on relevance judgment.

Kappa statistic is a common measure for agreement between judges.

Xi'an Jiaotong-Liverpool University
西交利物浦大學

Kappa Measure: Example

Number of docs	Judge 1	Judge 2
300	Relevant	Relevant
70	Nonrelevant	Nonrelevant
20	Relevant	Nonrelevant
10	Nonrelevant	Relevant

Kappa Measure: Example

Kappa = [P(A) – P(E)] / [1 – P(E)]

$P(A)$:proportion of the times judges agreed

= (300+70)/400 = 0.925

$P(E)$: the proportion of the times the judges would be expected to agree by chance.

Chance Agreement

There are choices in how P(E) is estimated:

- if we simply say we are making a two-class decision and assume nothing more, then the expected chance agreement rate is 0.5;

- With the given class distribution, usually *marginal statistics* is used to calculated P(E).

$$P(E) = P(nonrelevant)^2 + P(relevant)^2$$

pools the marginal distribution across judges.

Marginal Statistics

$P(1) = \frac{300+70}{400}$

$= 0.925$

$\therefore K = \frac{0.925 - 0.665}{1 - 0.665}$

$\simeq 0.776$

		Judge 2 Relevance		
		Yes	No	Total
Judge 1 Relevance	Yes	300	20	320
	No	10	70	80
	Total	310	90	400

For a contingency table, as above, a *marginal statistic* is formed by summing a row or column.

$P(E) = P^2(1r) + P^2(R)$

$= \left(\frac{90+80}{400\times 2}\right)^2 + \left(\frac{320+310}{400\times 2}\right)^2$

$\simeq 0.665$

Observed proportion of the times the judges agreed

$$P(A) = (300 + 70)/400 = 370/400 = 0.925$$

Pooled marginals

$$P(nonrelevant) = (80 + 90)/(400 + 400) = 170/800 = 0.2125$$

$$P(relevant) = (320 + 310)/(400 + 400) = 630/800 = 0.7878$$

Probability that the two judges agreed by chance

$$P(E) = P(nonrelevant)^2 + P(relevant)^2 = 0.2125^2 + 0.7878^2 = 0.665$$

Kappa statistic

$$\kappa = (P(A) - P(E))/(1 - P(E)) = (0.925 - 0.665)/(1 - 0.665) = 0.776$$

Kappa Measure: Example

Kappa measure

- Designed for categorical judgments
- Corrects for chance agreement P(E)

Kappa = [P(A) – P(E)] / [1 – P(E)]

Kappa = 0 is they agree only at the rate given by chance,
1 for total agreement(two judges always agree).

Kappa Measure: Example

P(nonrelevant) = (10+20+70+70)/800 = 0.2125

P(relevant) = (10+20+300+300)/800 = 0.7878

Probability that two judges agreed by chance:

P(E) = ?

Kappa ?

Kappa Measure: Example

P(nonrelevant) = (10+20+70+70)/800 = 0.2125

P(relevant) = (10+20+300+300)/800 = 0.7878

Probability that two judges agreed by chance:

P(E) = 0.2125^2 + 0.7878^2 = 0.665

Kappa = (0.925 – 0.665)/(1-0.665) = 0.776

Kappa Example

Kappa > 0.8: good agreement;

0.67 < Kappa < 0.8: acceptable;

Kappa < 0.67: need redesign relevance assessment methodology;

Depends on purpose of study

For >2 judges: an average pairwise kappa value.

Relevance Feedback and Query Expansion

Improving Recall

Options for improving recall:

- Local methods: Do a “local” on-demand analysis for a user query.

  Main local method: **relevance feedback**;

- Global methods: Do a global analysis for the collection to produce **thesaurus**

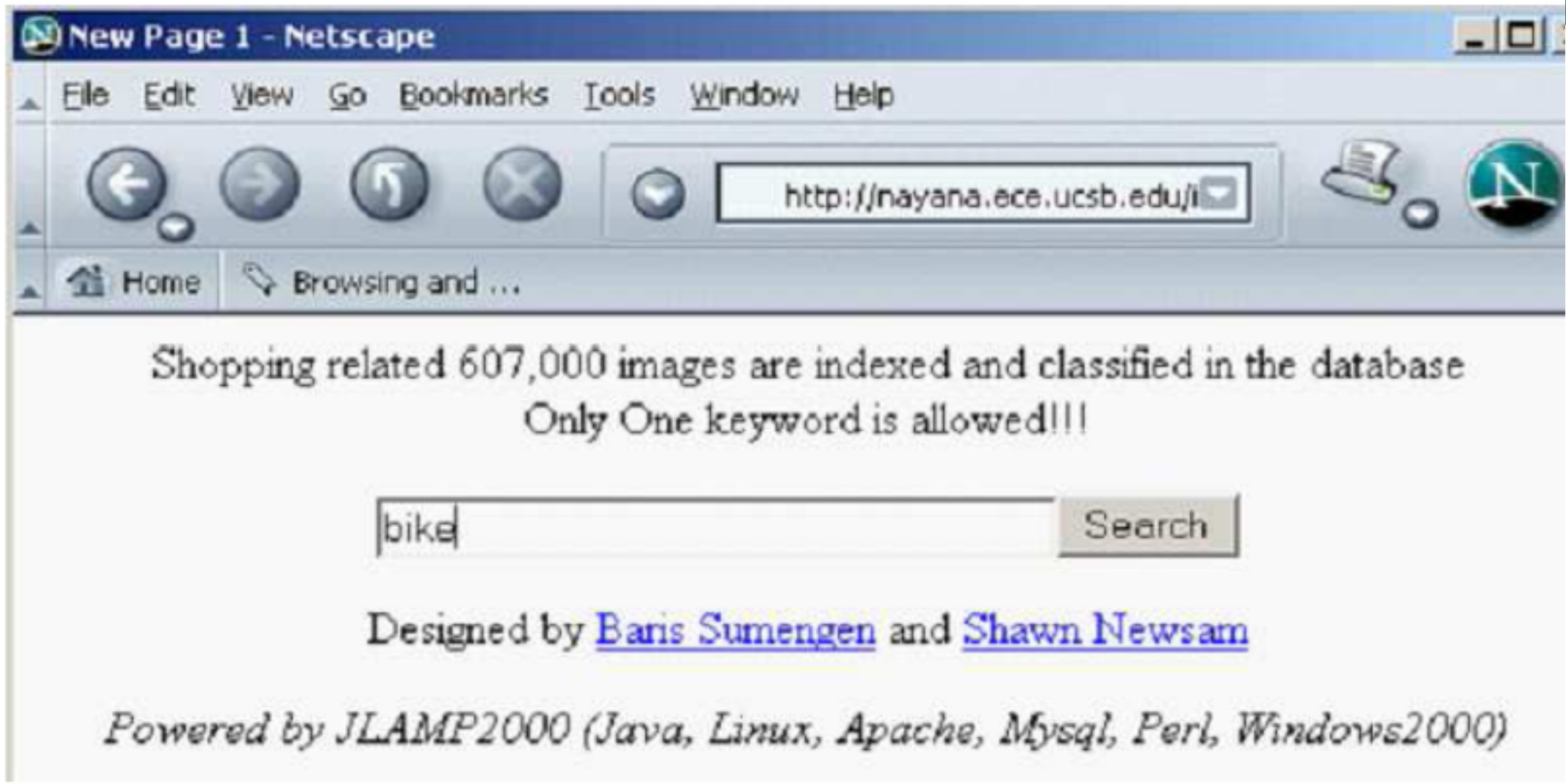
  Use thesaurus for **query expansion.**

Relevance feedback

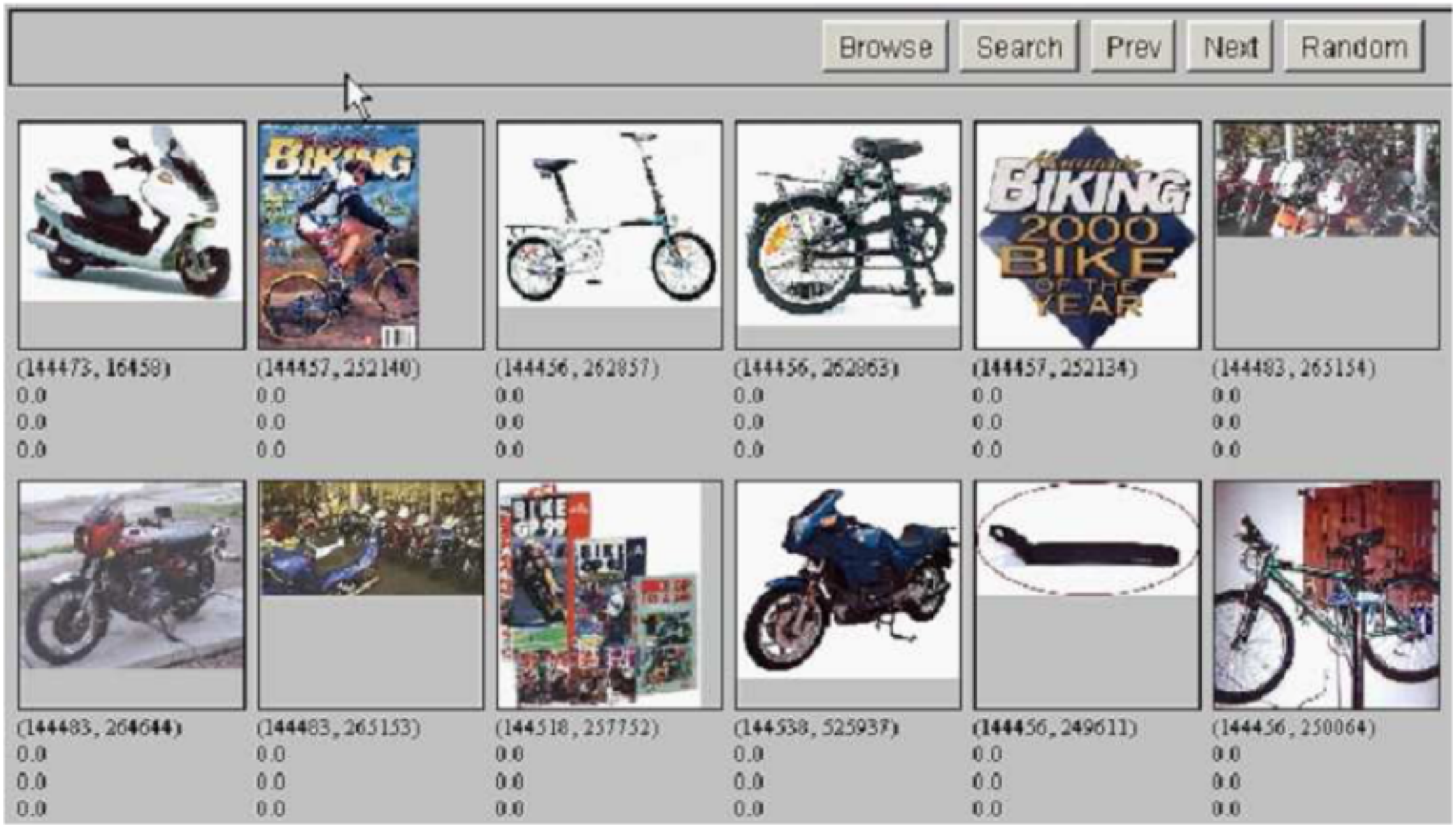
The idea of ***relevance feedback*** **(RF)** is to involve the user in the retrieval process so as to improve the final result set.

The basic procedure is:

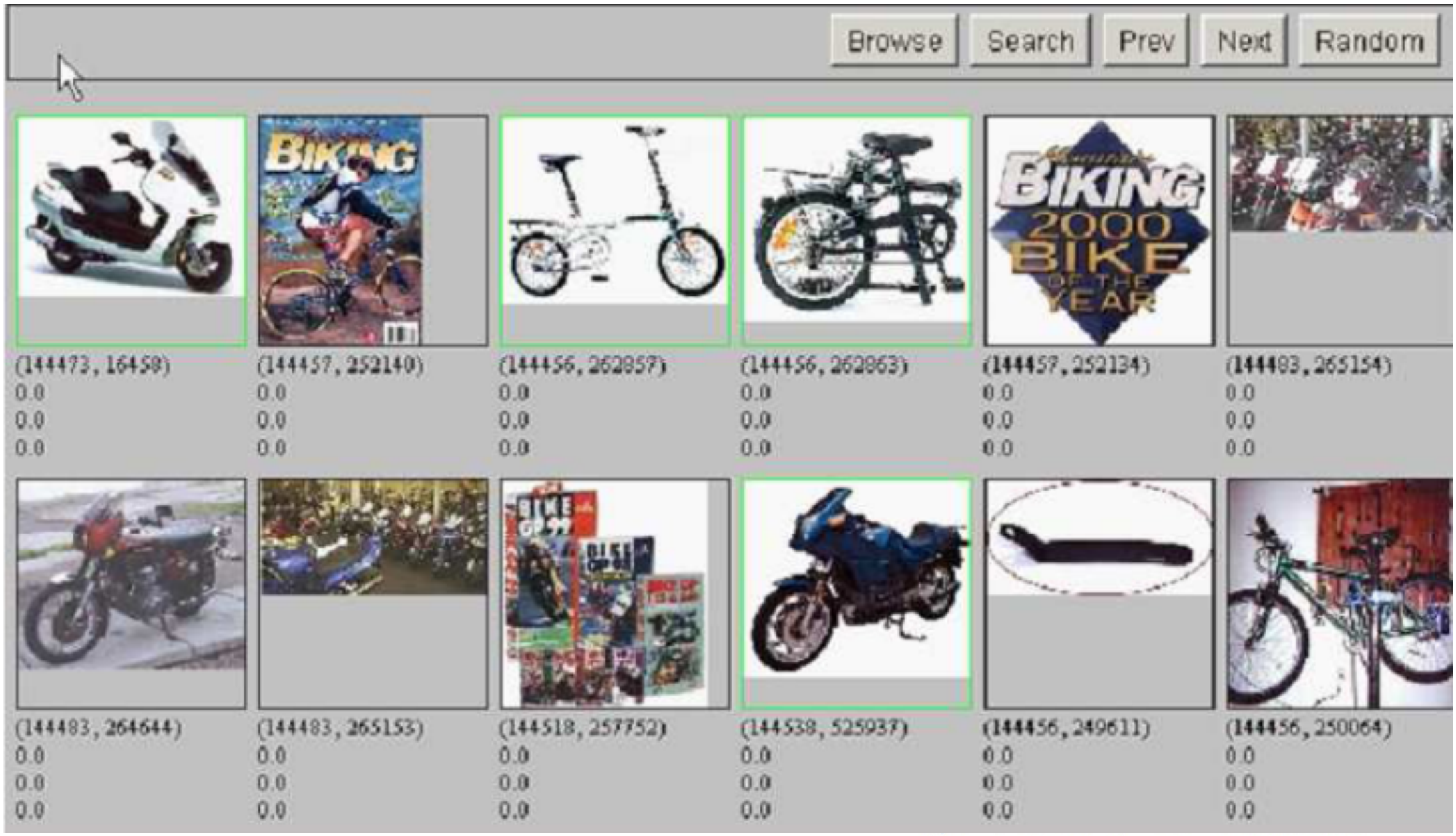
1. The user issues a (short, simple) query.
2. The system returns an initial set of retrieval results.
3. The user marks some returned documents as relevant or nonrelevant.
4. The system computes a better representation of the information need based on the user feedback.
5. The system displays a revised set of retrieval results.


New Page 1 - Netscape
File Edit View Go Bookmarks Tools Window Help
http://nayana.ece.ucsb.edu/i
Home
Browsing and ...
Shopping related 607,000 images are indexed and classified in the database
Only One keyword is allowed!!!
bike
Search
Designed by Baris Sumengen and Shawn Newsam
Powered by JLAMP2000 (Java, Linux, Apache, Mysql, Perl, Windows2000)

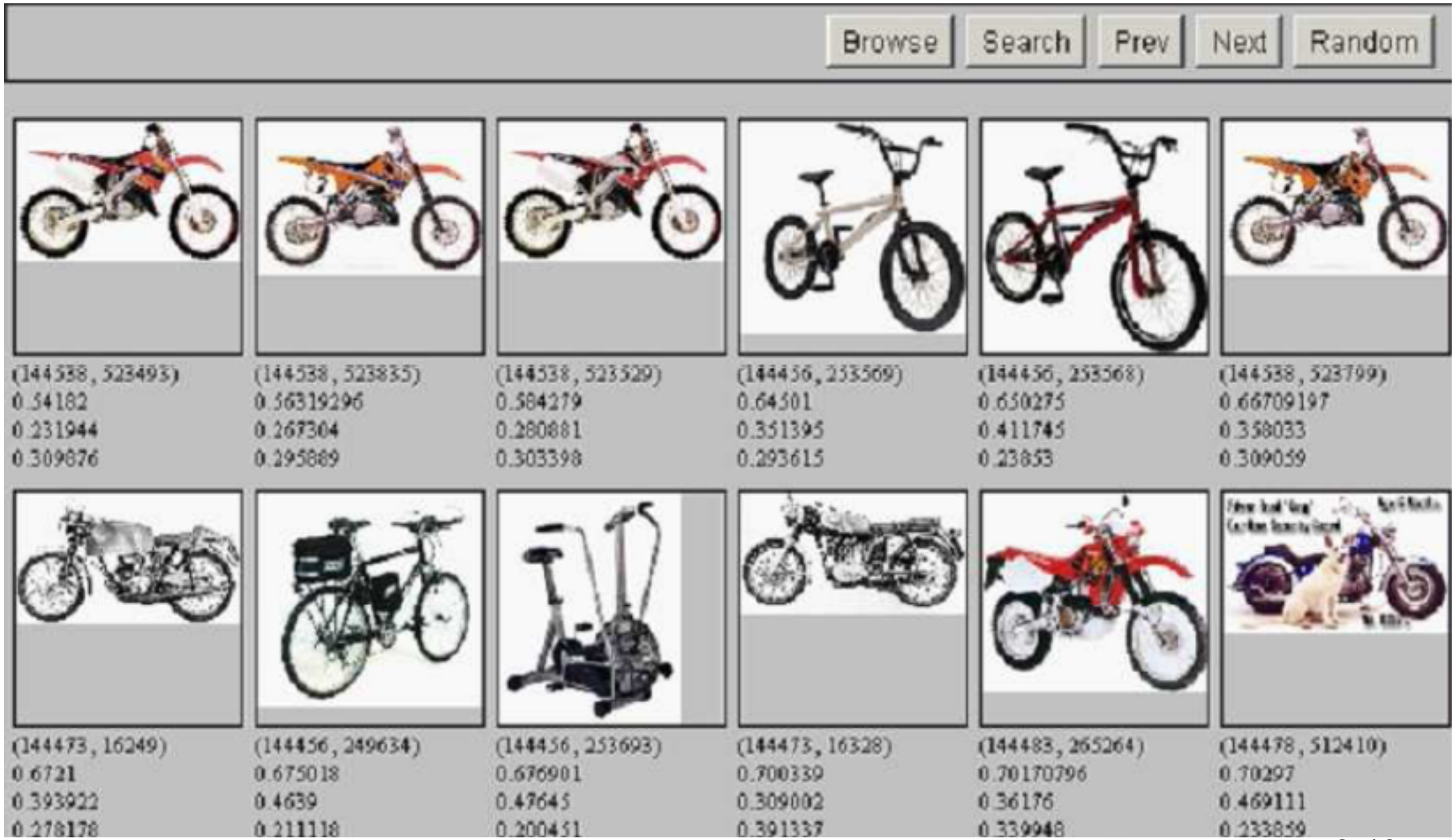
Results for initial query



Select what is relevant



Results after relevance feedback



Relevance feedback

Image search provides a good example of relevance feedback.

This is a domain where a user can easily have difficulty formulating what they want in words, but can easily indicate relevant or nonrelevant.

The underlying theory

Relevance feedback: refine a query with user's input.

To refine a query, firstly we need to know how to define the best query.

The optimal query vector will maximizes similarity with relevant documents while minimizing similarity with non-relevant documents.

Xi'an Jiaotong-Liverpool University
西交利物浦大學

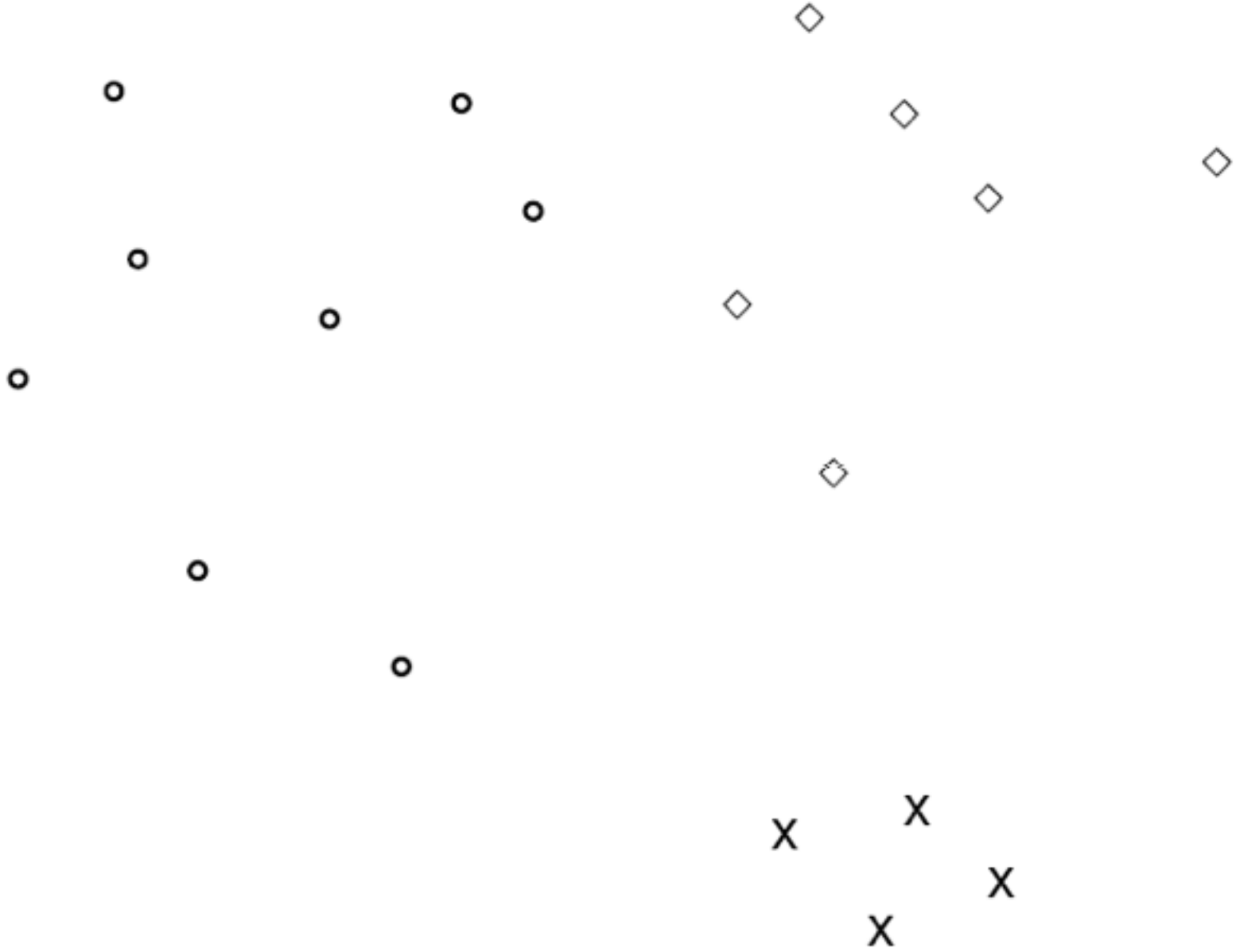
Key concept for relevance feedback: Centroid

- 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:

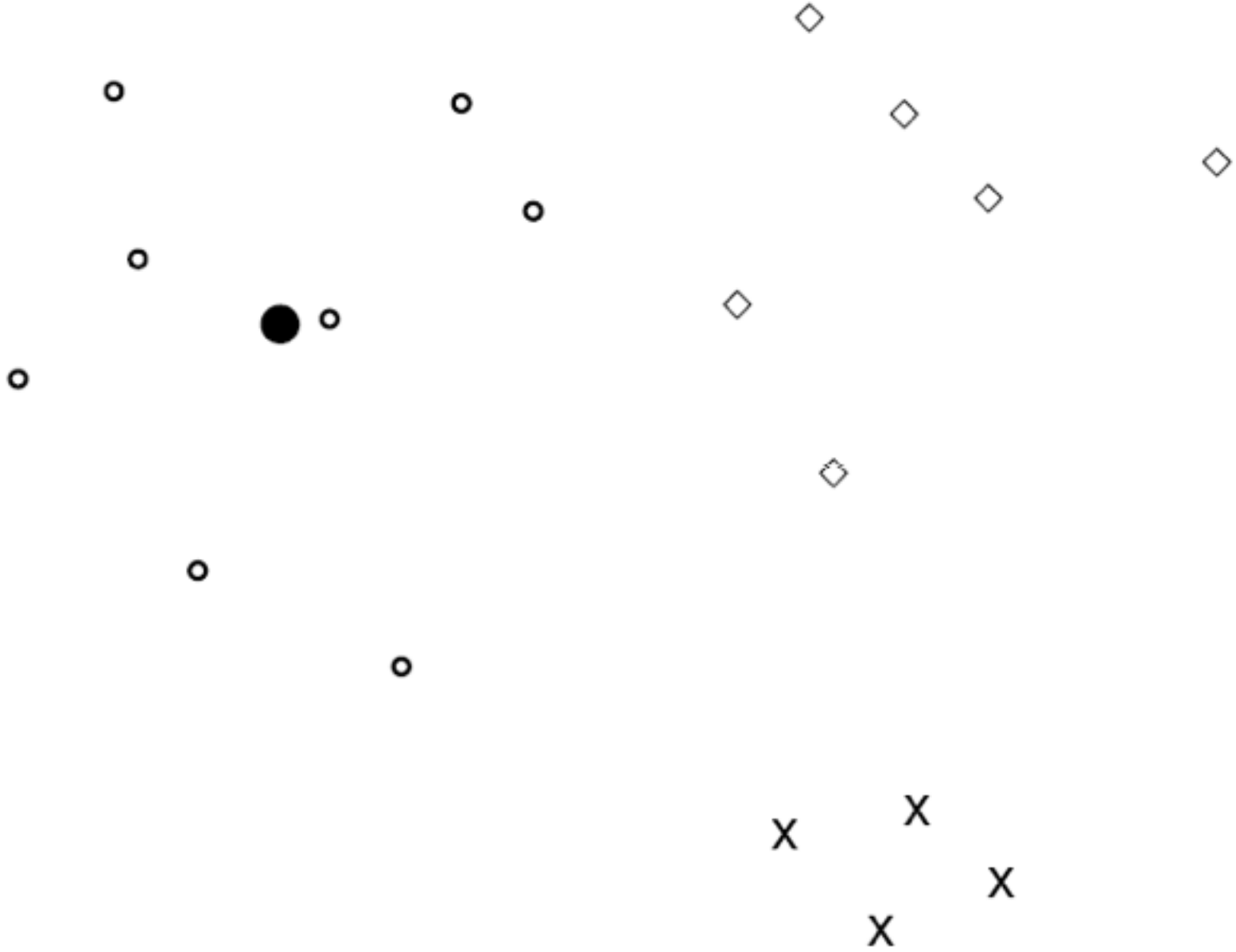
$$\vec{\mu}(D) = \frac{1}{|D|} \sum_{d \in D} \vec{v}(d)$$

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

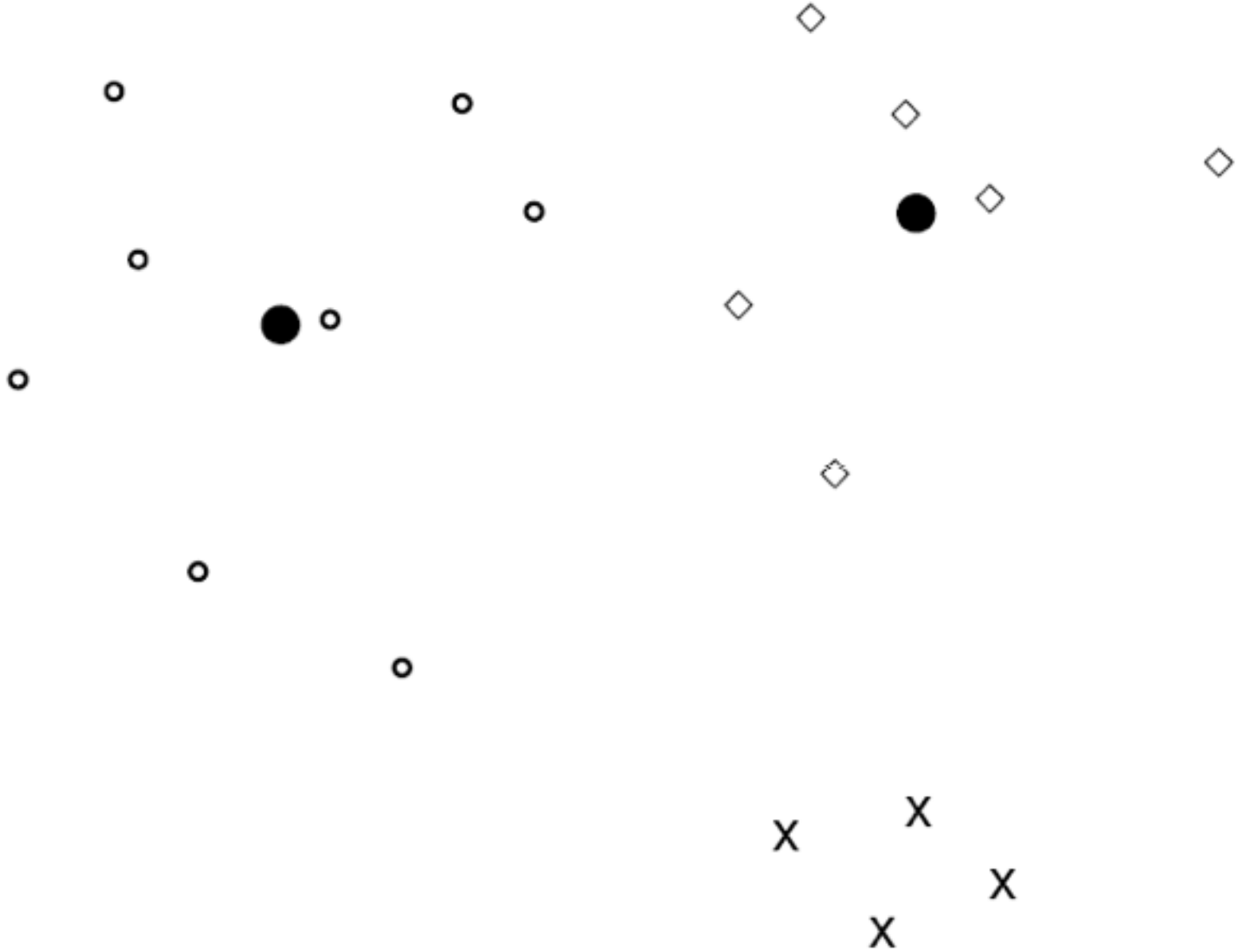
Centroid: Examples



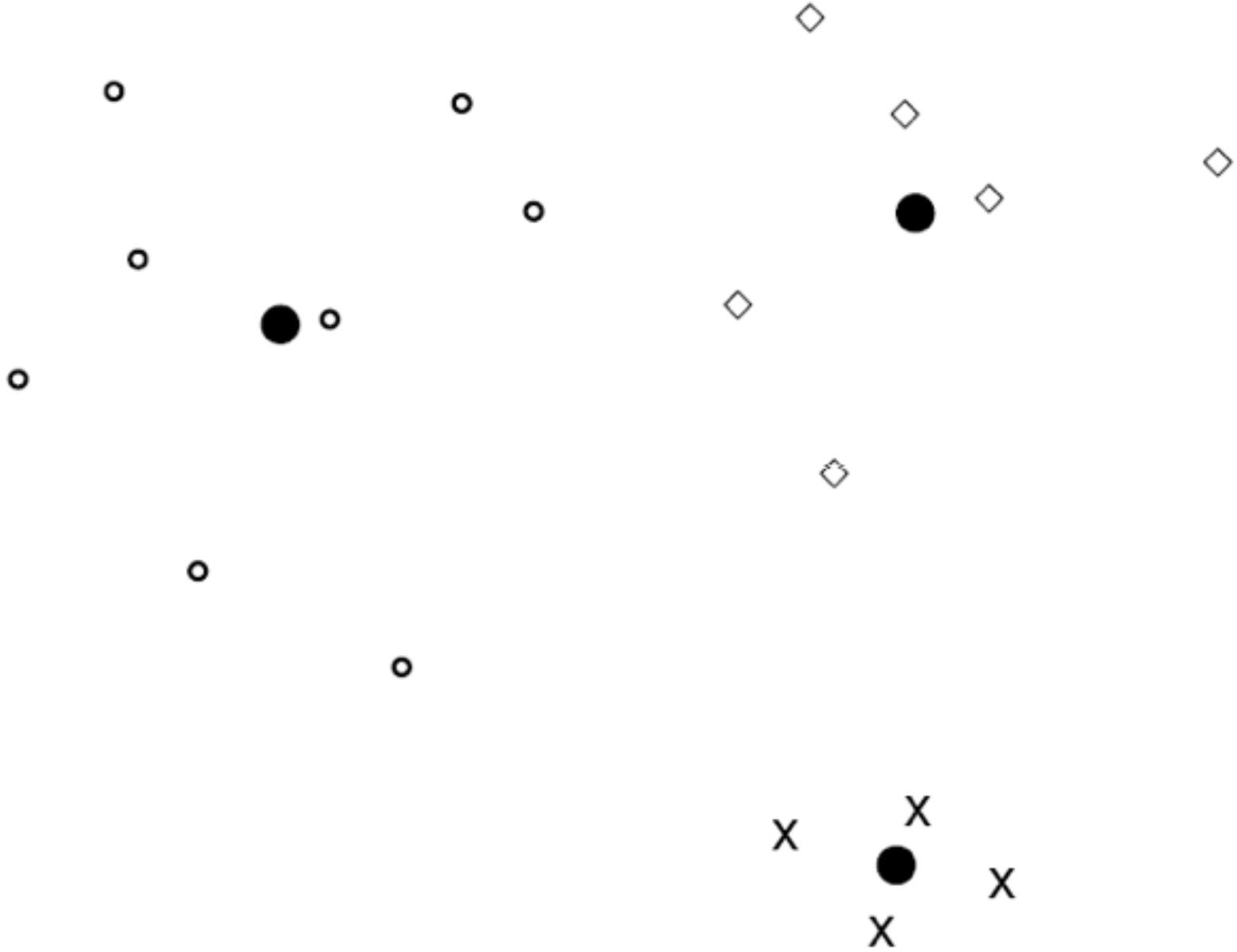
Centroid: Examples



Centroid: Examples



Centroid: Examples



Rocchio algorithm

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

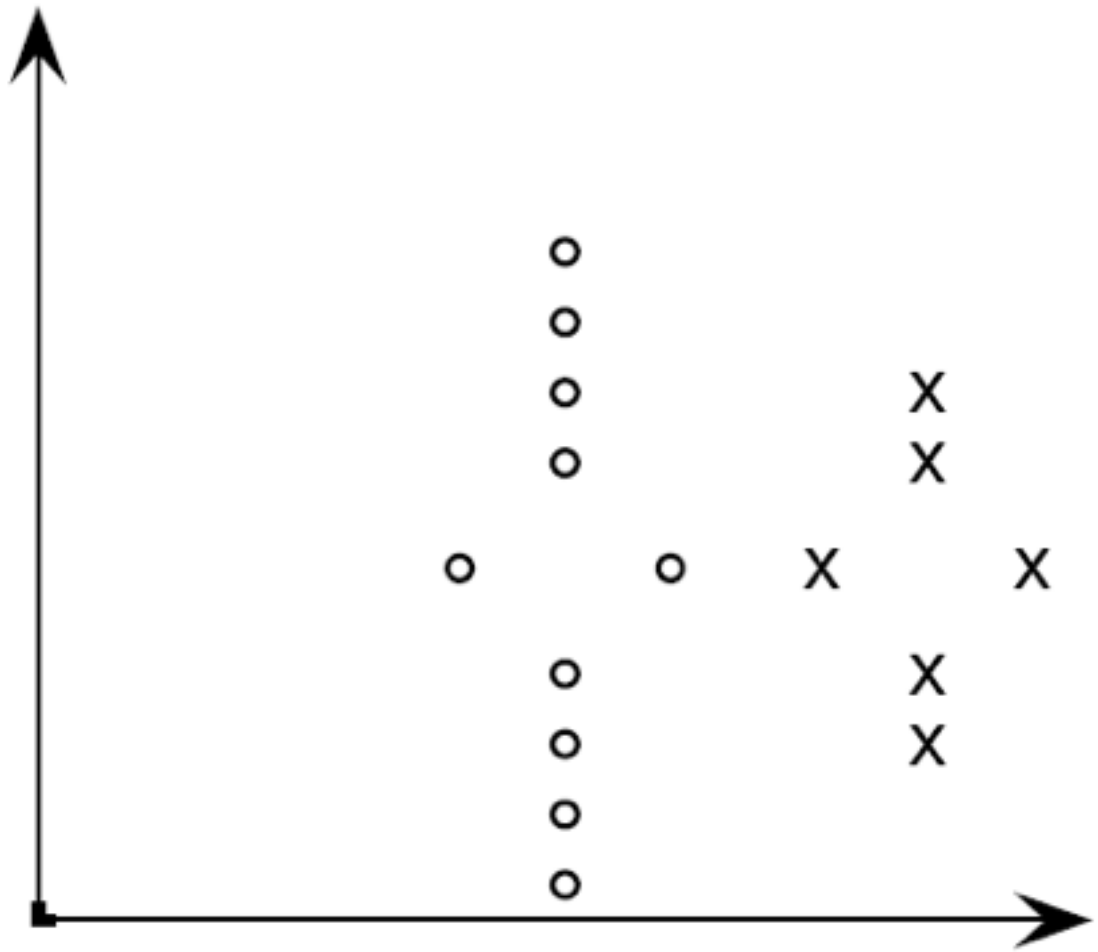
$$\vec{q}_{opt} = \arg\max_{\vec{q}}[\text{sim}(\vec{q}, D_r) - \text{sim}(\vec{q}, D_{nr})]$$

- Closely related to maximum separation between relevant and nonrelevant docs
- This optimal query vector is:

$$\vec{q}_{opt} = \frac{1}{|D_r|}\sum_{\vec{d}_j \in D_r}\vec{d}_j + [\frac{1}{|D_r|}\sum_{\vec{d}_j \in D_r}\vec{d}_j - \frac{1}{|D_{nr}|}\sum_{\vec{d}_j \in D_{nr}}\vec{d}_j]$$

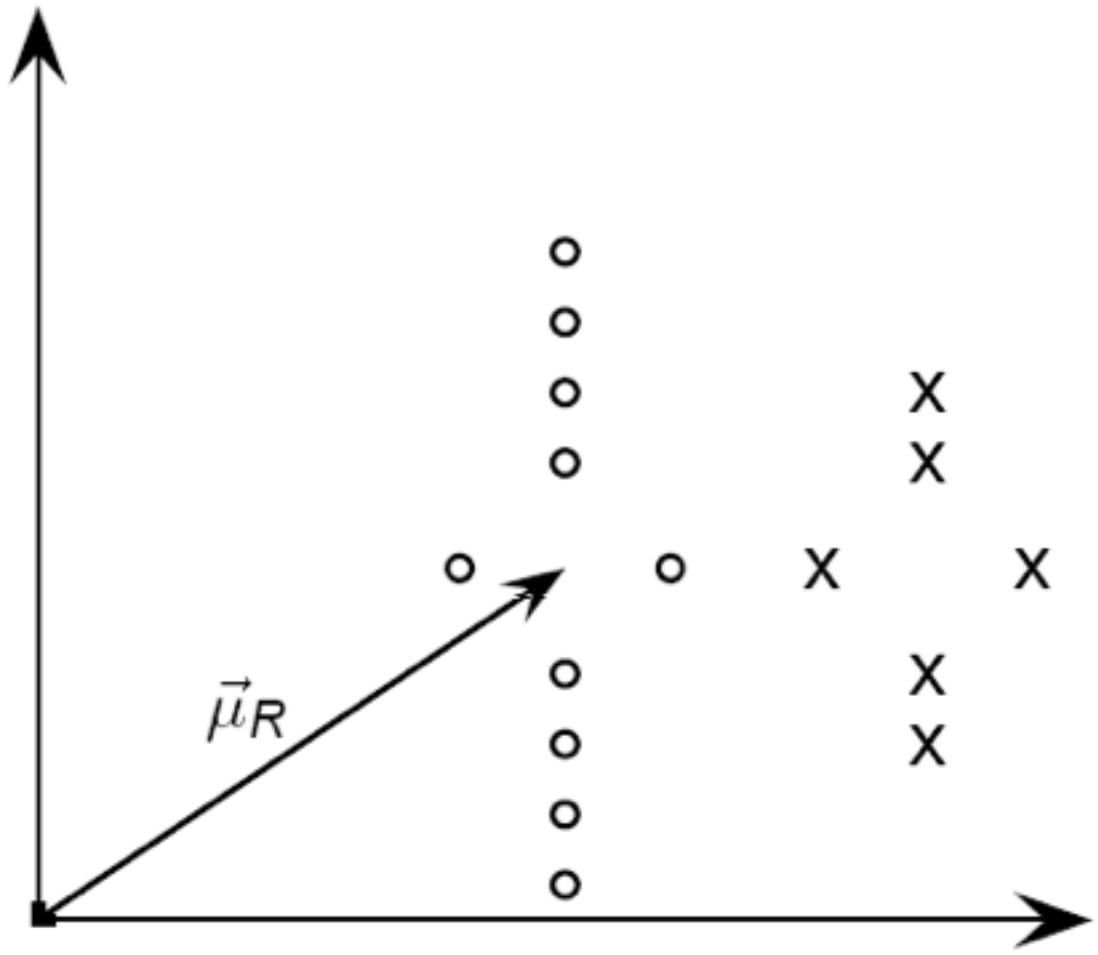
D_r: set of relevant docs; D_{nr}: set of nonrelevant docs

Rocchio illustrated



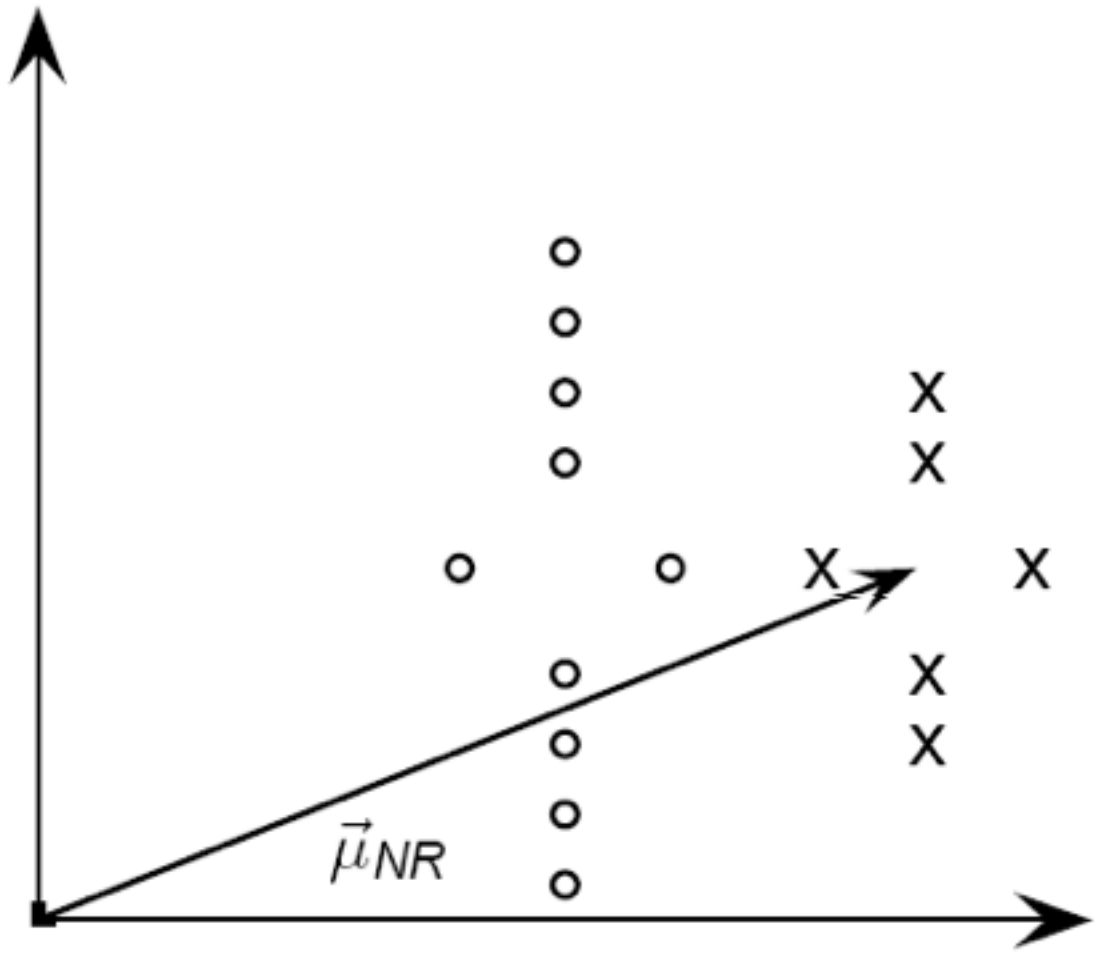
circles: relevant documents, Xs: nonrelevant documents

Rocchio illustrated



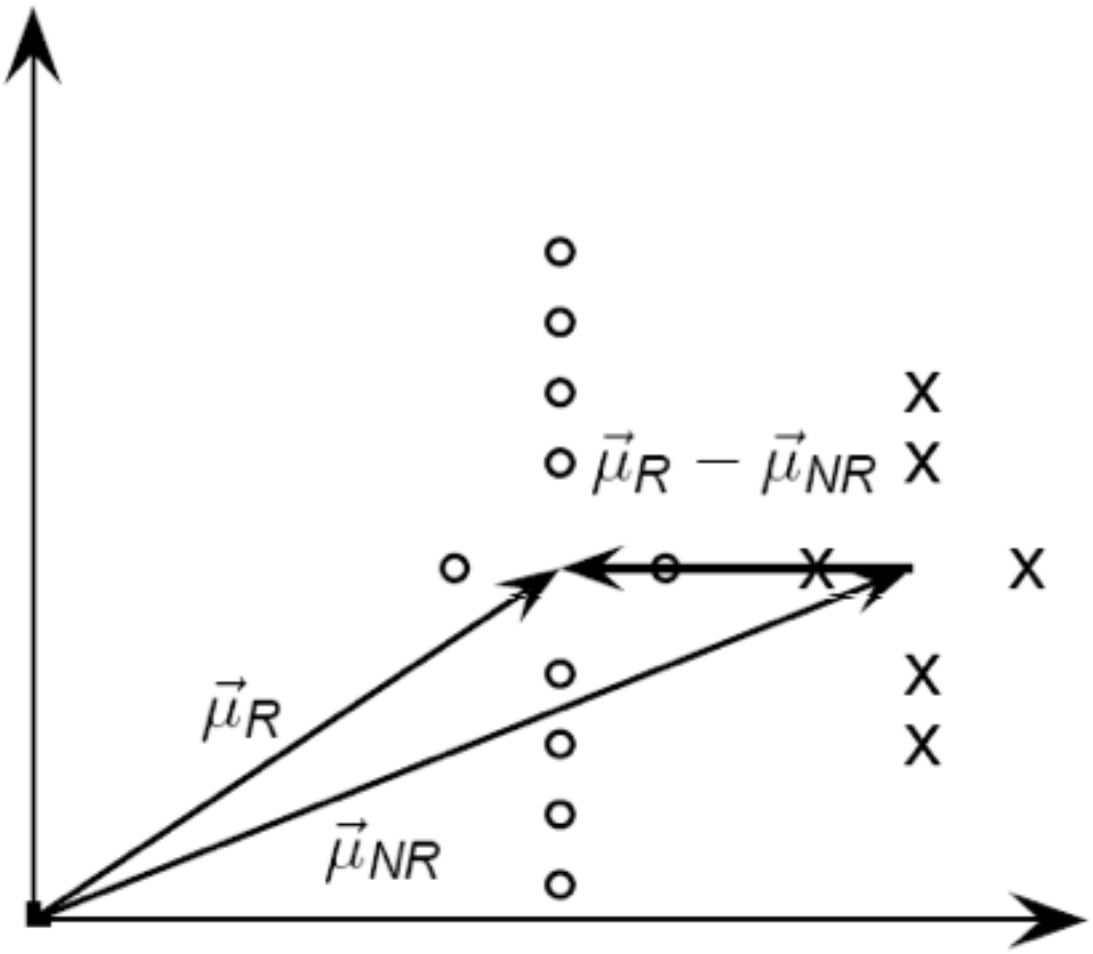
$\vec{\mu}_R$: centroid of relevant documents

Rocchio illustrated



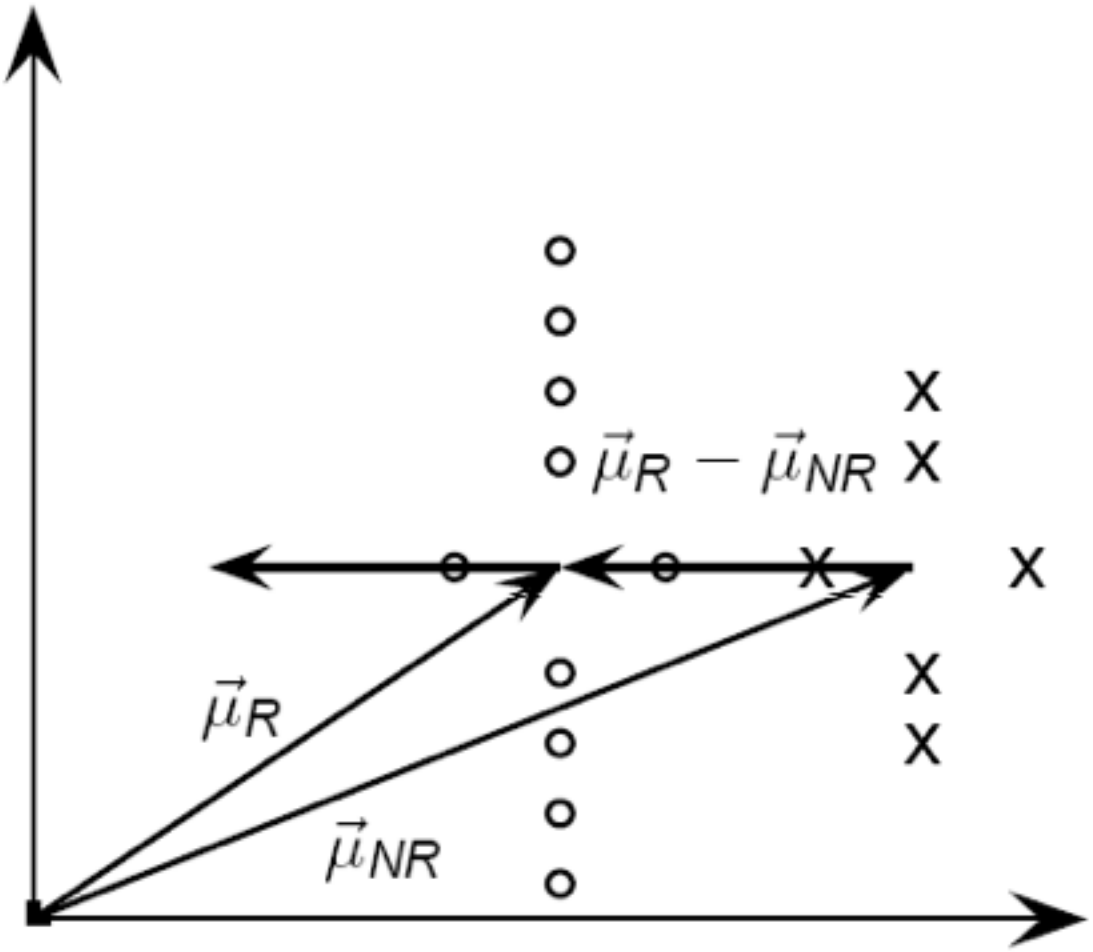
$\vec{\mu}_{NR}$: centroid of nonrelevant documents

Rocchio illustrated



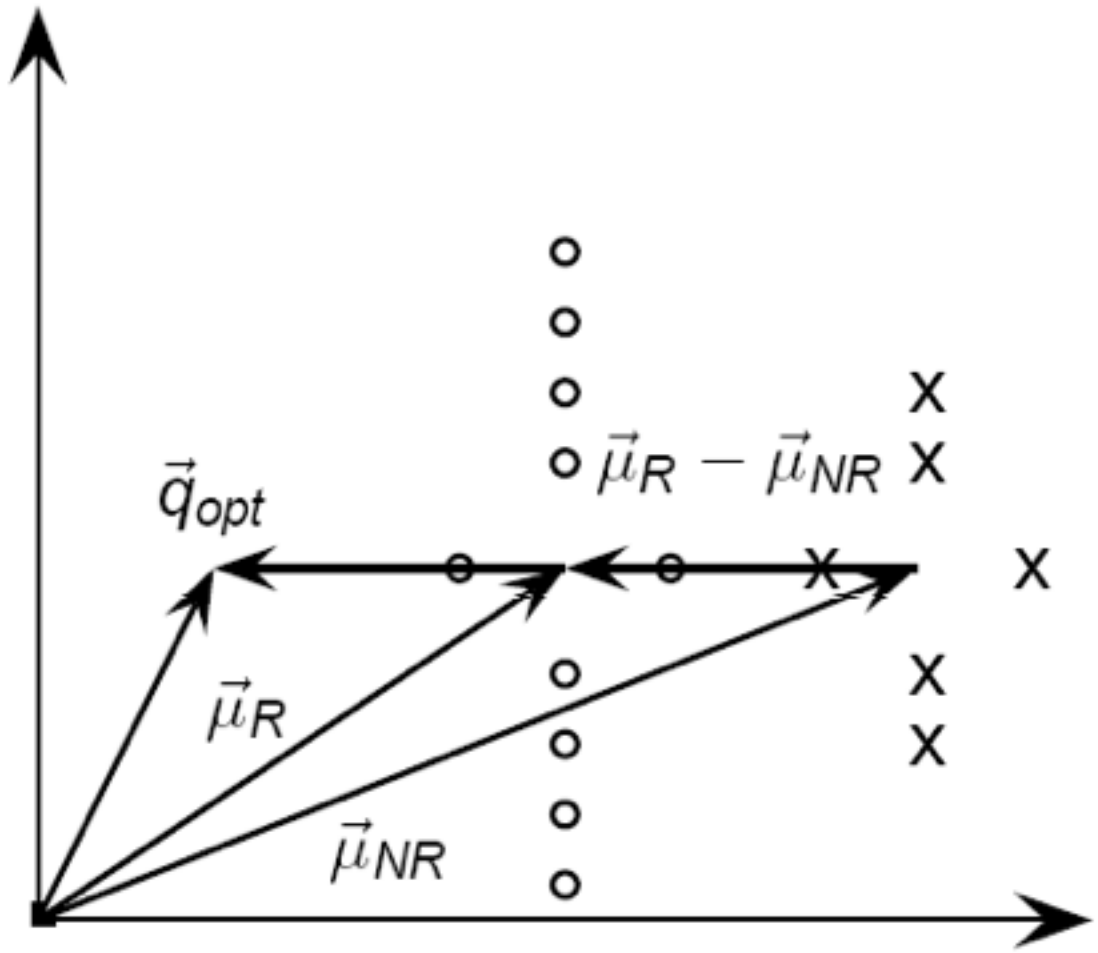
$\vec{\mu}_R - \vec{\mu}_{NR}$: difference vector

Rocchio illustrated



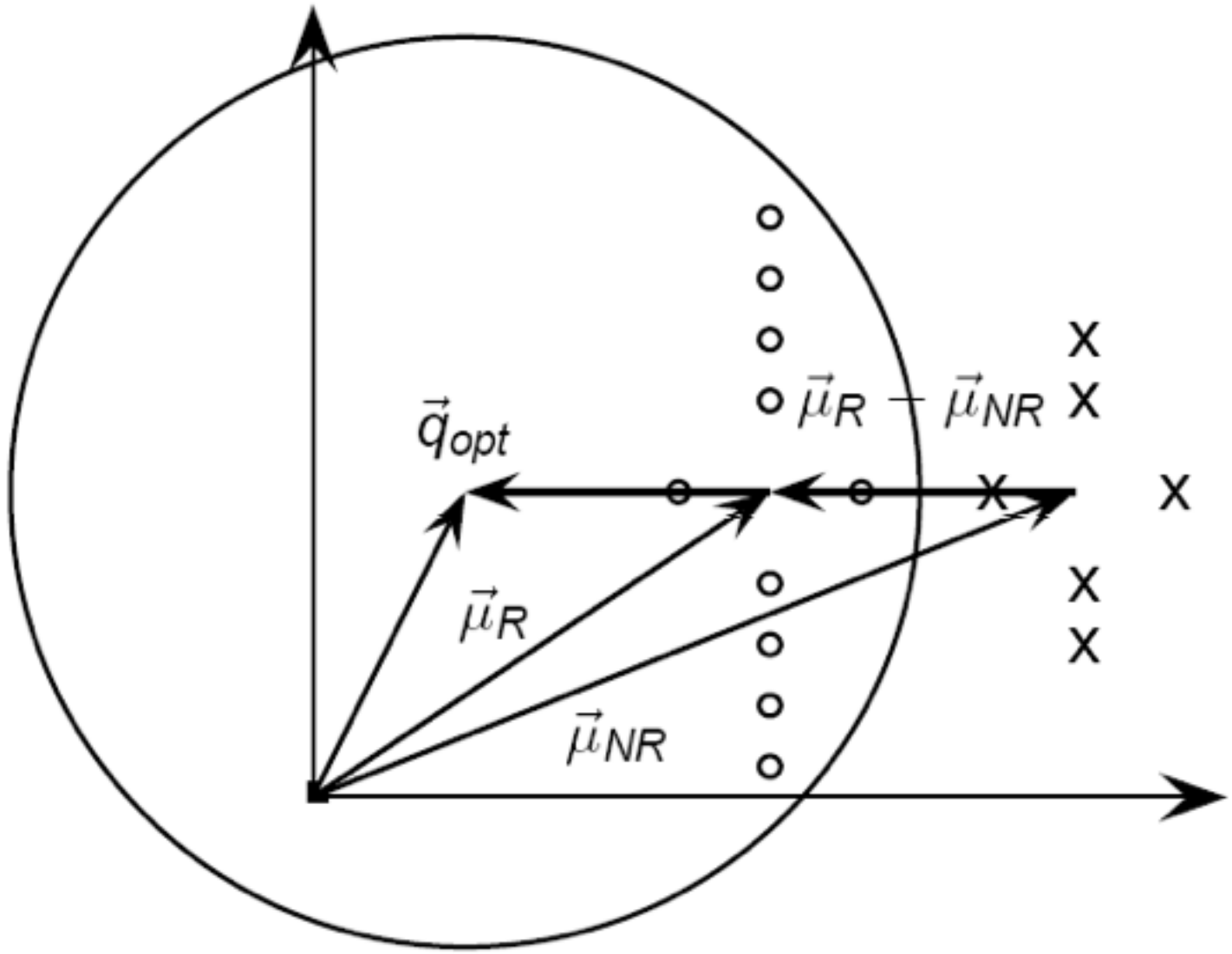
Add difference vector to $\vec{\mu}_R$ ...

Rocchio illustrated



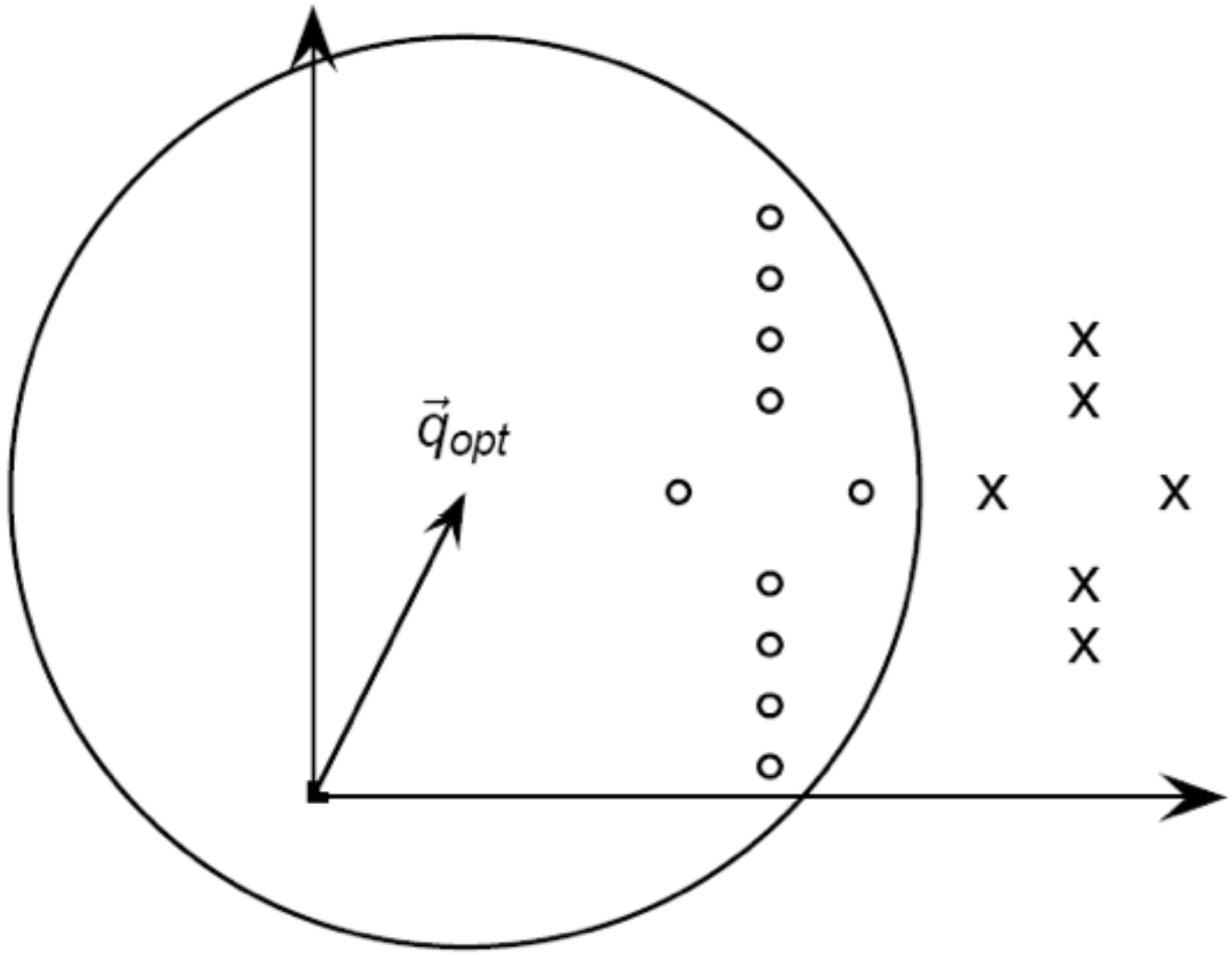
. . . to get $\vec{q}_{opt}$

Rocchio illustrated



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

Rocchio illustrated



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

Rocchio 1971 algorithm (SMART)

- Used in practice:

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

  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 attached to each term
- 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.

Exercise II

Suppose that a user's initial query is cheap CDs cheap DVDs extremely cheap CDs. The user examines two documents, *d*1 and *d*2. She judges *d*1, with the content *CDs cheap software cheap CDs* relevant and *d*2 with content *cheap thrills DVDs* nonrelevant. Assume that we are using direct term frequency (with no scaling and no document frequency). There is no need to length-normalize vectors. Using Rocchio relevance feedback, what would the revised query vector be after relevance feedback? Assume α= 1, β = 0.75, γ = 0.25.

3.5 4.25 0.75 1 0.75 0

SOLUTION. Term frequencies:

word	query q	d_1	d_2
CDs	2	2	0
cheap	3	2	1
DVDs	1	0	1
extremely	1	0	0
software	0	1	0
thrills	0	0	1

For $1.0 * \vec{q} + 0.75 * \vec{d} + 1 - 0.25 * \vec{d}_2$, we get: $(3.5\ \ 4.25\ \ 0.75\ \ 1\ \ 0.75\ \ -0.25)^T$ or $(7/2\ \ 17/4\ \ 3/4\ \ 1\ \ 3/4\ \ -1/4)^T$. Negative weights are set to 0. The Rocchio vector thus is: $(3.5\ \ 4.25\ \ 0.75\ \ 1\ \ 0.75\ \ 0)^T$.

Query Expansion

Query Expansion

The most common form of query expansion is global analysis, using some form of **thesaurus**.

For each term *t* in a query, the query can be automatically expanded with synonyms and related words of *t* from the thesaurus.

Building up a thesaurus

Three methods:

1. Use of a controlled vocabulary that is maintained by human editors.

2. An automatically derived thesaurus. Word co-occurrence statistics over a collection of documents are used to automatically induce a thesaurus.

3. Query-equivalence based on query log mining.