Introduction to

Information Retrieval

Lecture 10: 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

❶ Motivation

❷ Relevance feedback: Basics

❸ Relevance feedback: Details

❹ 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”

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

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 an example of relevance feedback.

Example: 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

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		scientist

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

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:

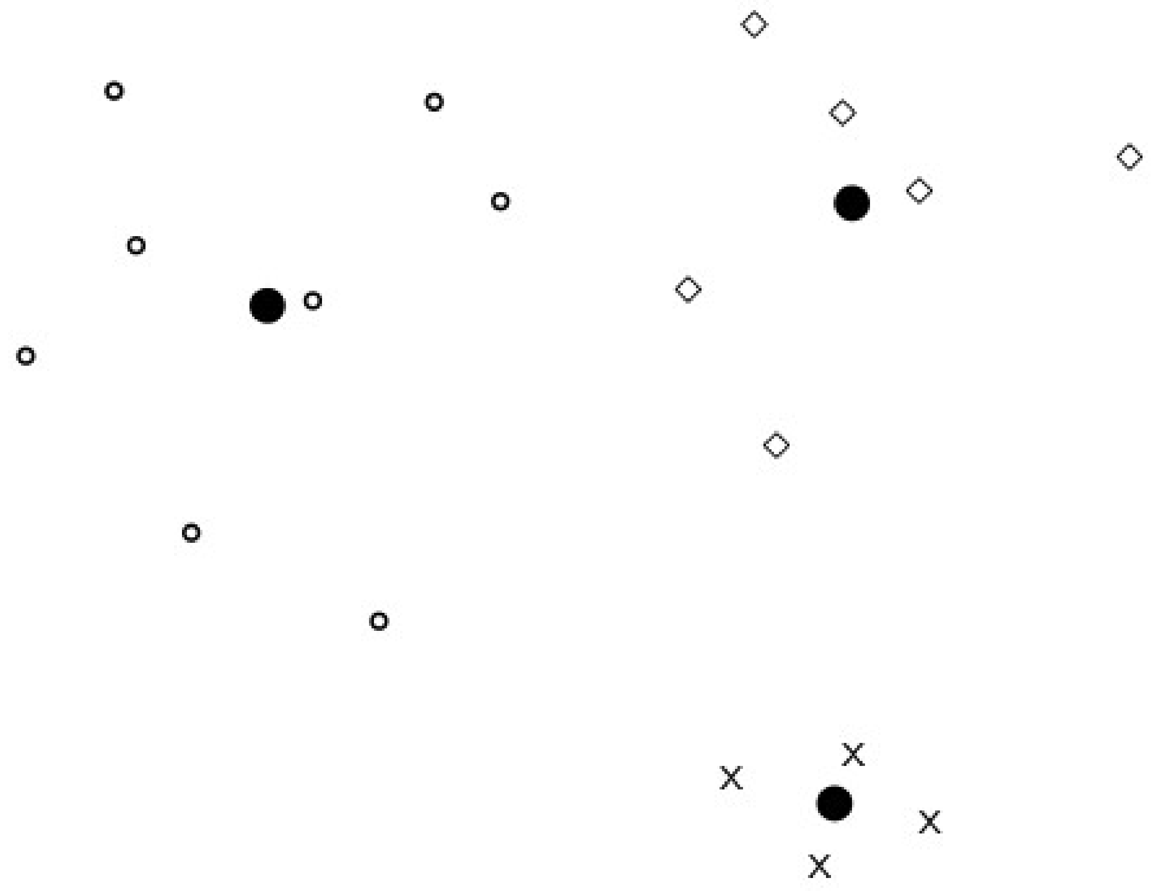
$$\vec{\mu}(D) = \frac{1}{|D|} \sum_{d \in D} \vec{v}(d)$$

$$\vec{v}(d) = \vec{d}$$

where D is a set of documents and

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 $\vec{q}_{opt}$ relevant and nonrelevant docs maximally.
- Making some additional assumptions, we can rewrite $\vec{q}_{opt}$ as:

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

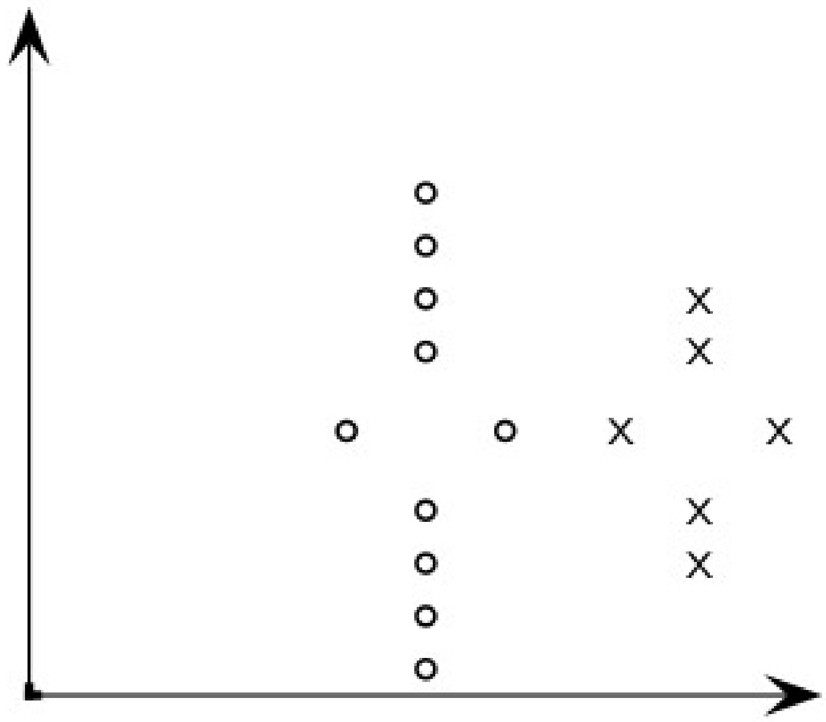
Rocchio' algorithm

- The optimal query vector is:

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

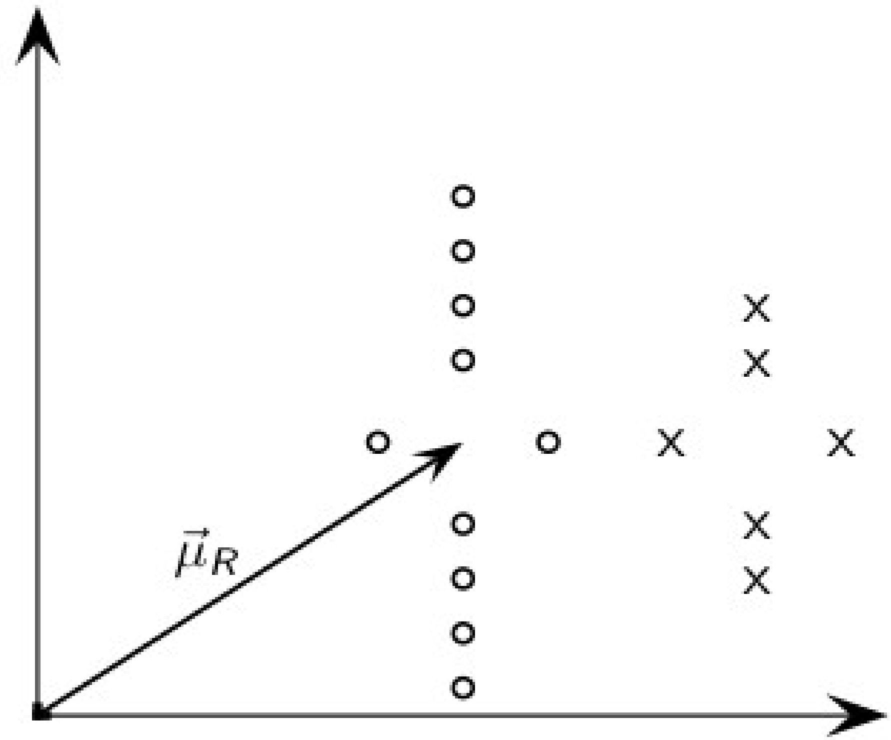
- We move the centroid of the relevant documents by the difference between the two centroids.

Exercise: Compute Rocchio' vector



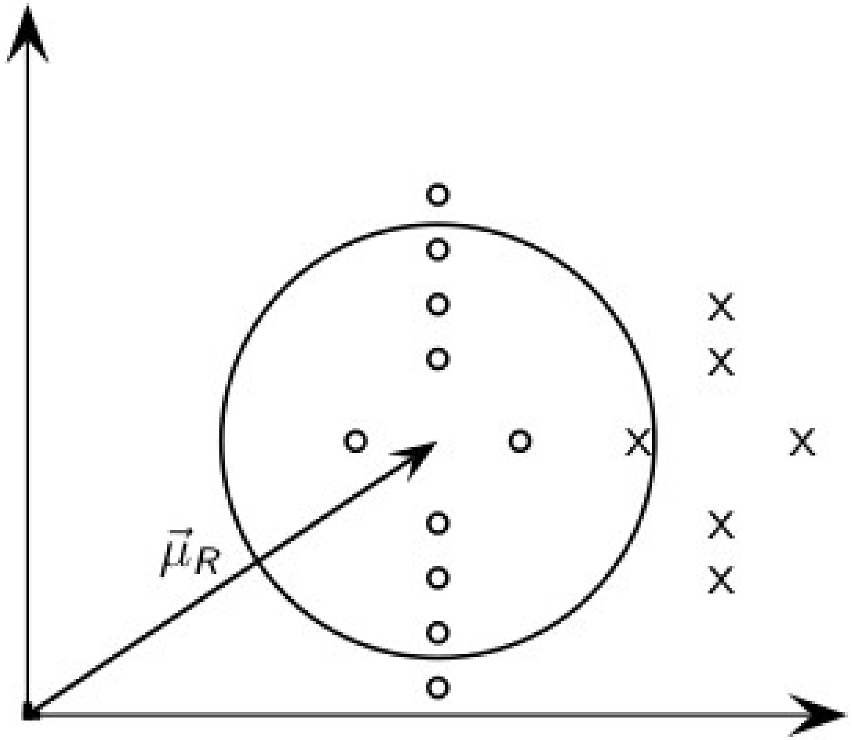
circles: relevant documents, Xs: nonrelevant documents

Rocchio' illustrated



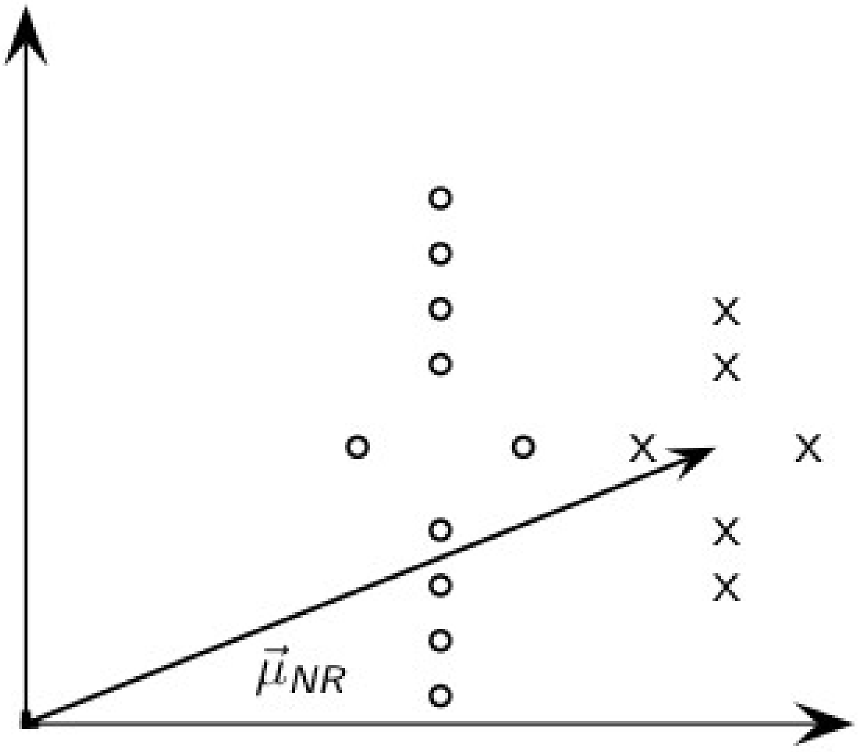
$\vec{\mu}_R$: centroid of relevant documents

Rocchio' illustrated



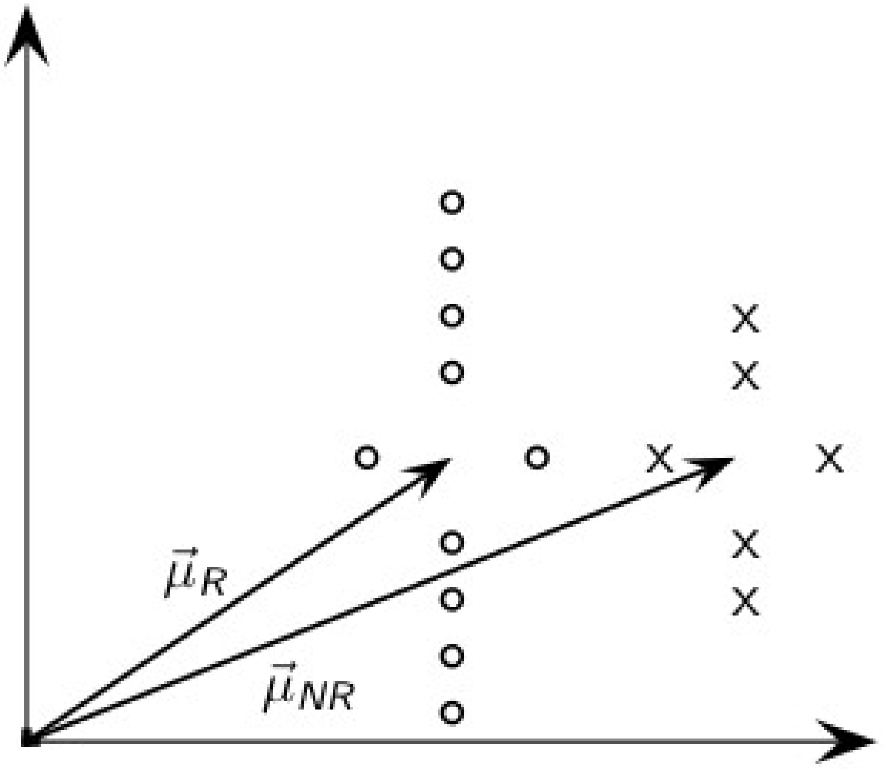
$\vec{\mu}_R$ does not separate relevant / nonrelevant.

Rocchio' illustrated

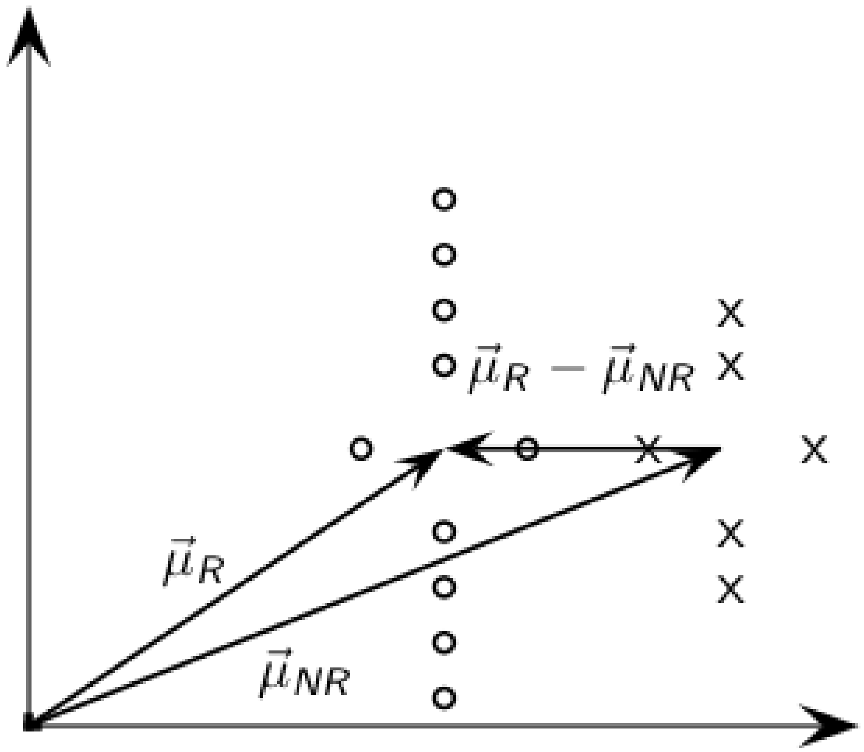


$\vec{\mu}_{NR}$: centroid of nonrelevant documents.

Rocchio’ illustrated

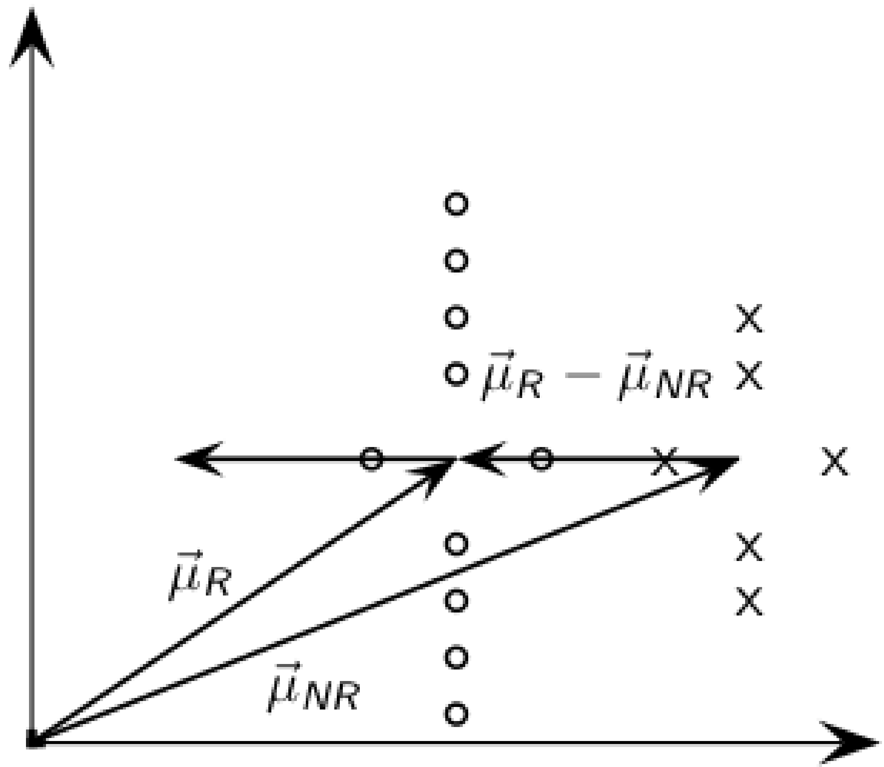


Rocchio' illustrated



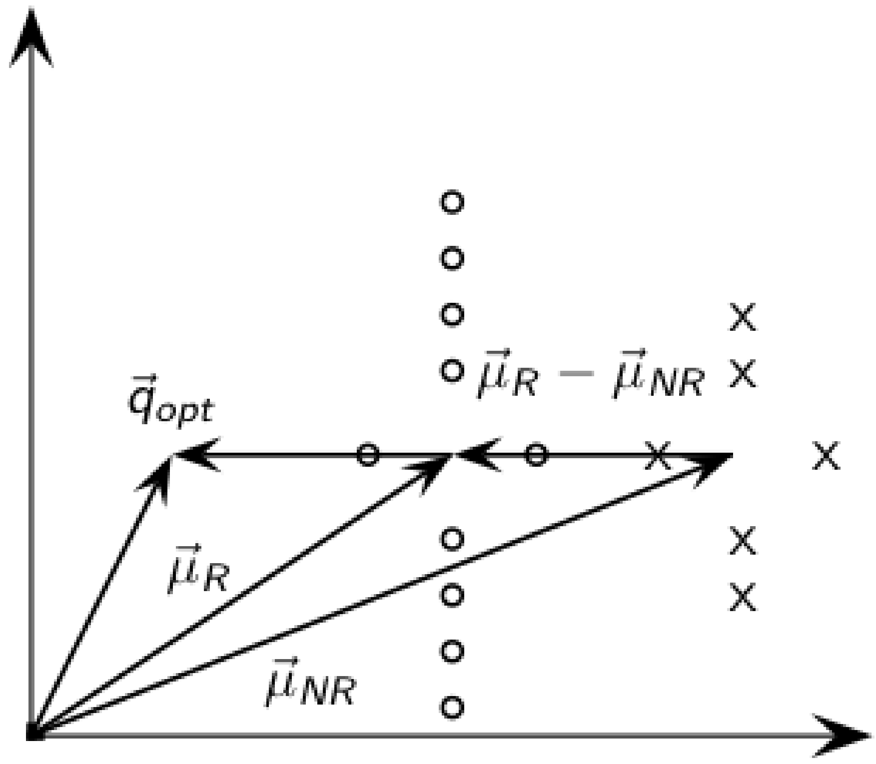
$\vec{\mu}_R \quad \vec{\mu}_{NR}$: difference vector

Rocchio' illustrated



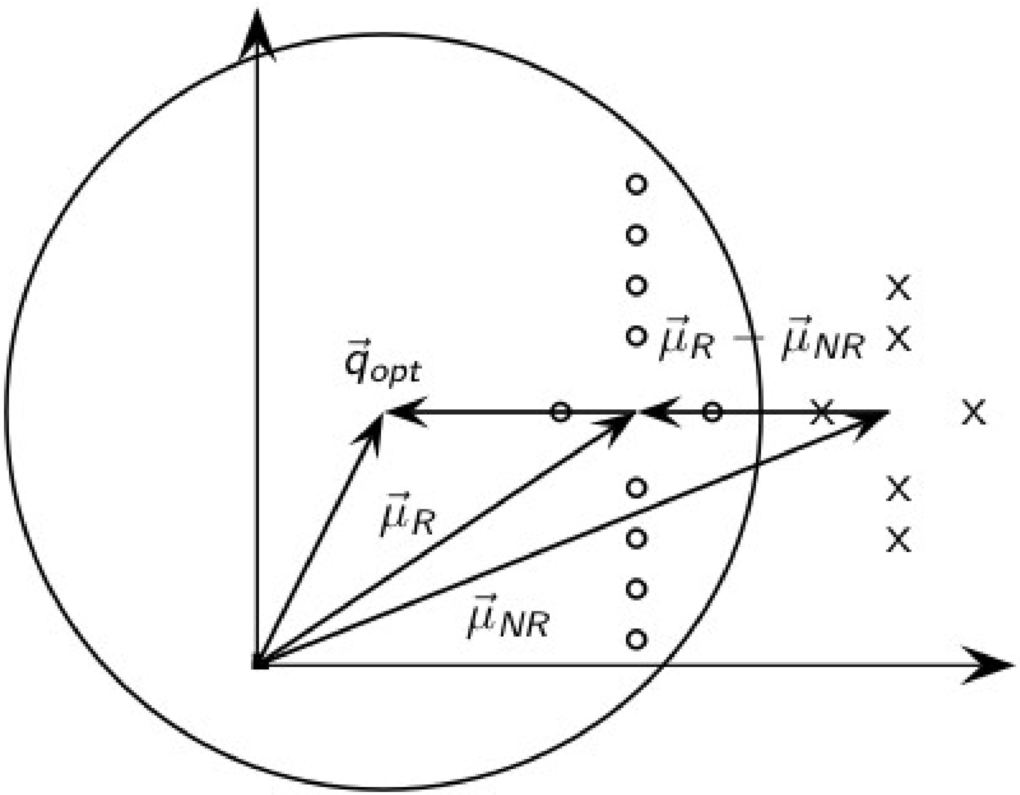
Add difference ve $\vec{\mu}_R$ to ...

Rocchio' illustrated



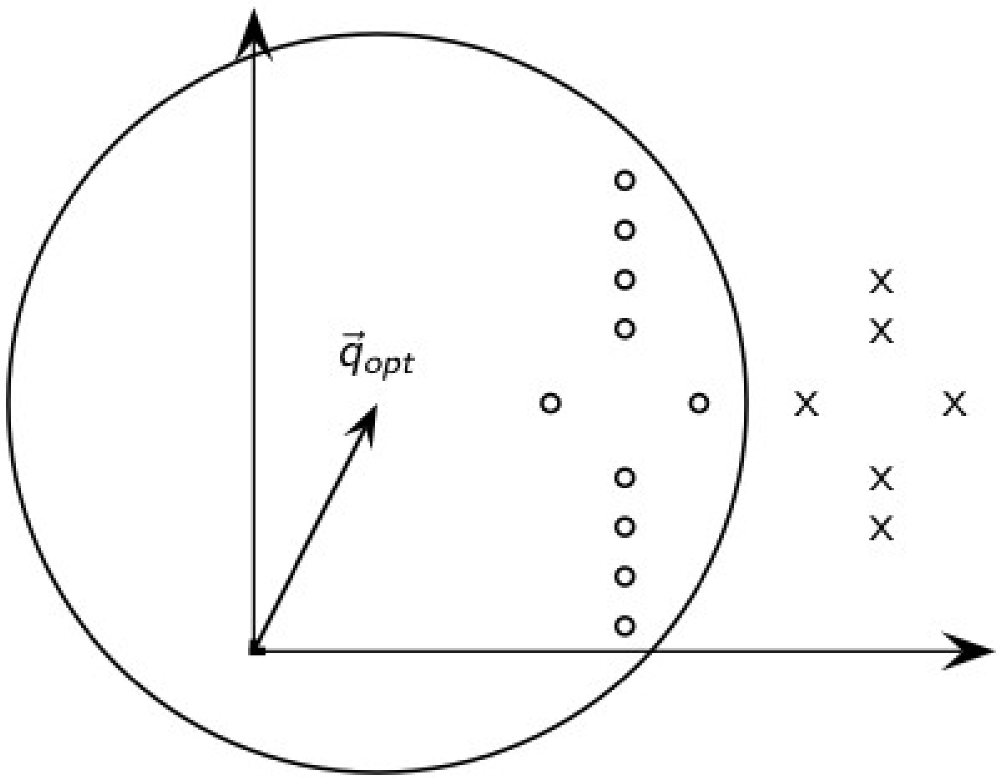
… to $\vec{q}_{opt}$

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

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.