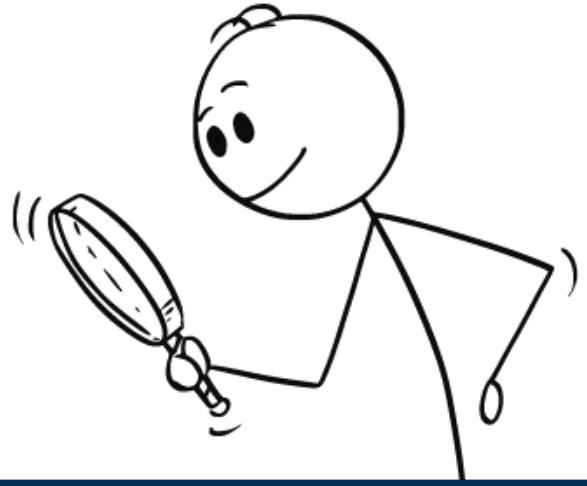


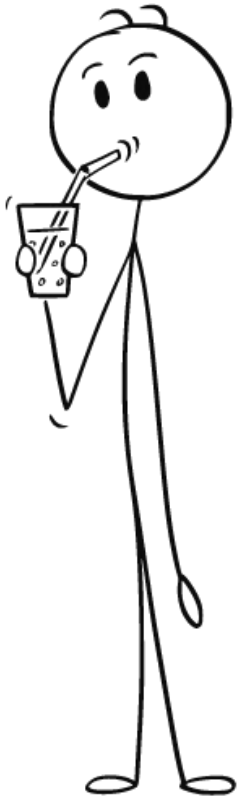
CSE 4/535
Information Retrieval

Sayantan Pal
PhD Student, Department of CSE
338Z Davis Hall

University at Buffalo
Department of CSE

Before we start

1. Midterm 2 Poll, Please answer by Tonight (Oct 18)
2. Timeline will be updated based on Poll responses, check website on Friday (Oct 20th)
3. Final Project: Upto 3 members will be allowed
4. Today's lecture - Relevance Feedback & Query Expansion



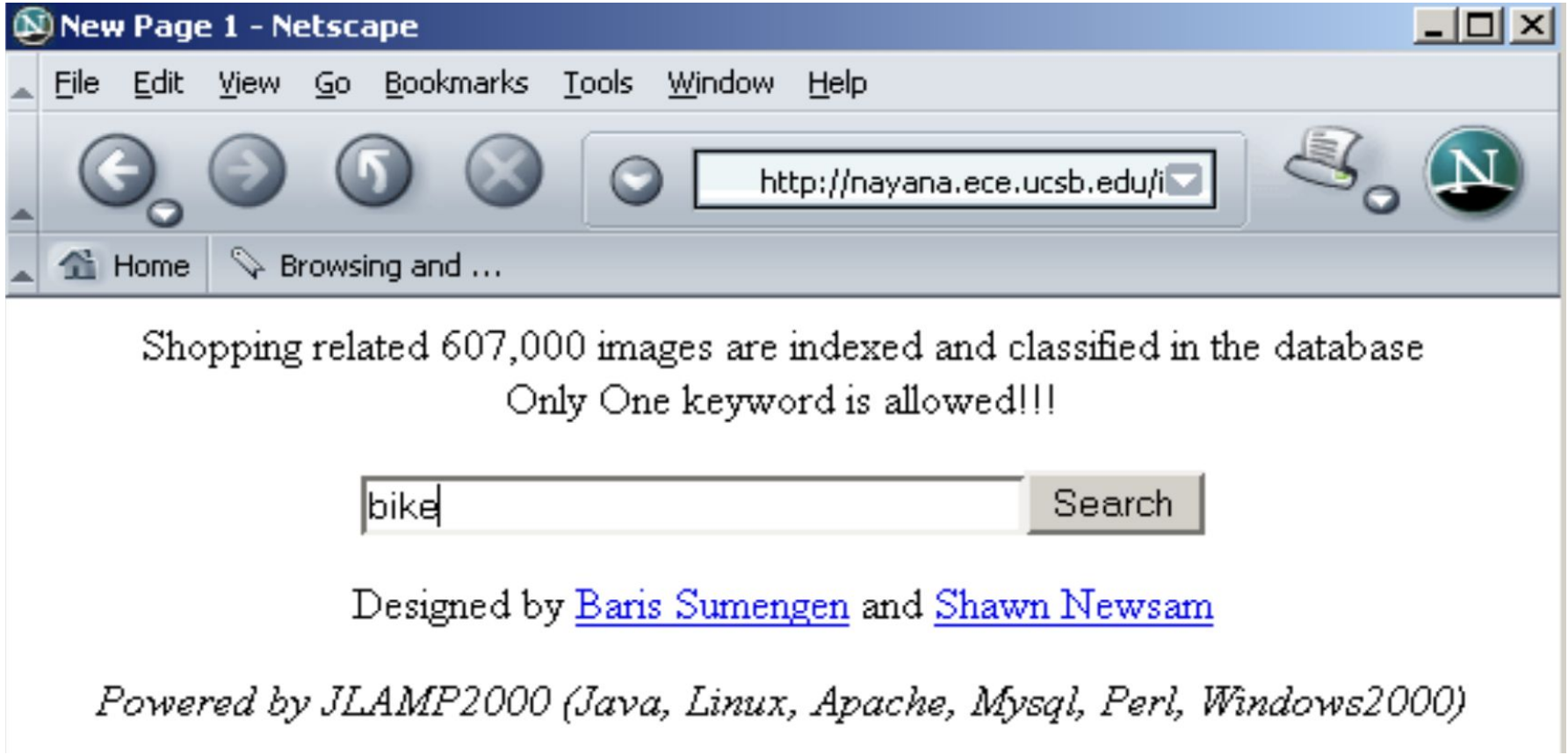
Recap - Previous Class

1. Evaluating a Search system
    a. MAP score
    b. Kappa Measure
    c. DCG, NDCG
    d. A/B Testing

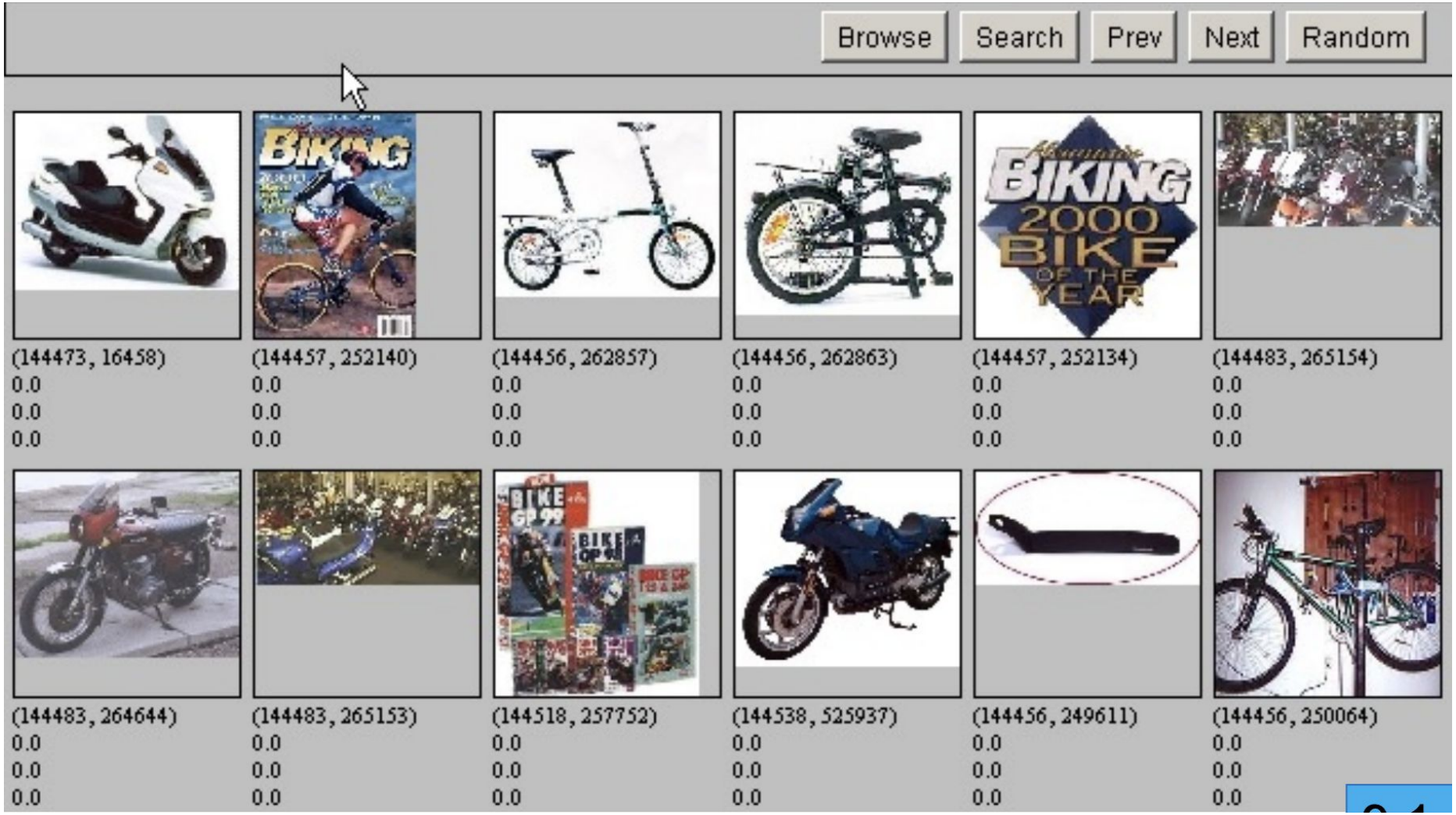
Relevance Feedback

- Relevance feedback: user feedback on relevance of docs in initial set of results
  - User issues a (short, simple) query
  - The user marks **some results** as relevant or non-relevant.
  - The system computes a better representation of the information need based on feedback.
  - Relevance feedback can go through **one or more** iterations.
- Idea: it may be difficult to formulate a good query when you don't know the collection well, so iterate

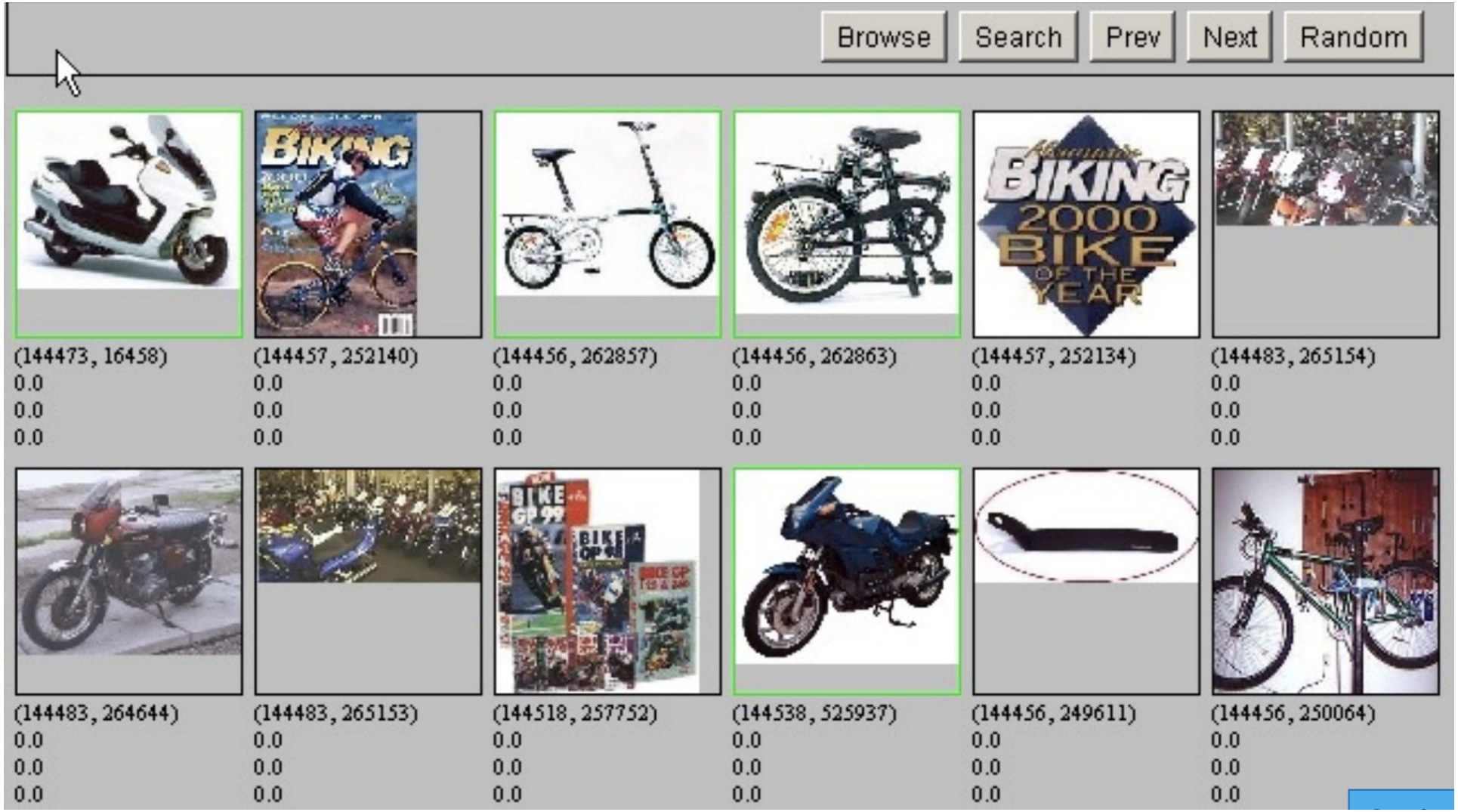
Relevance Feedback Example



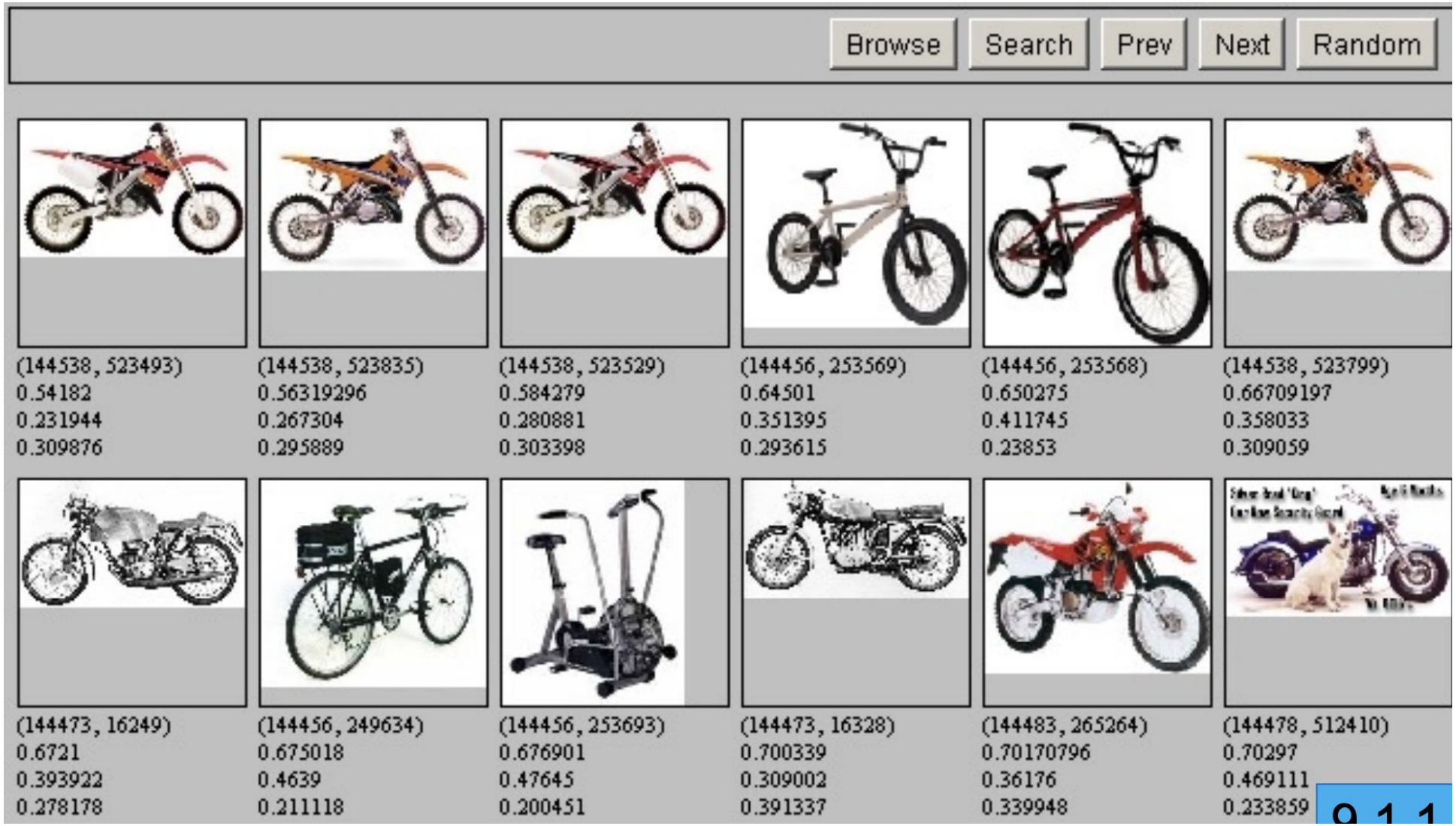
Results for Initial Query



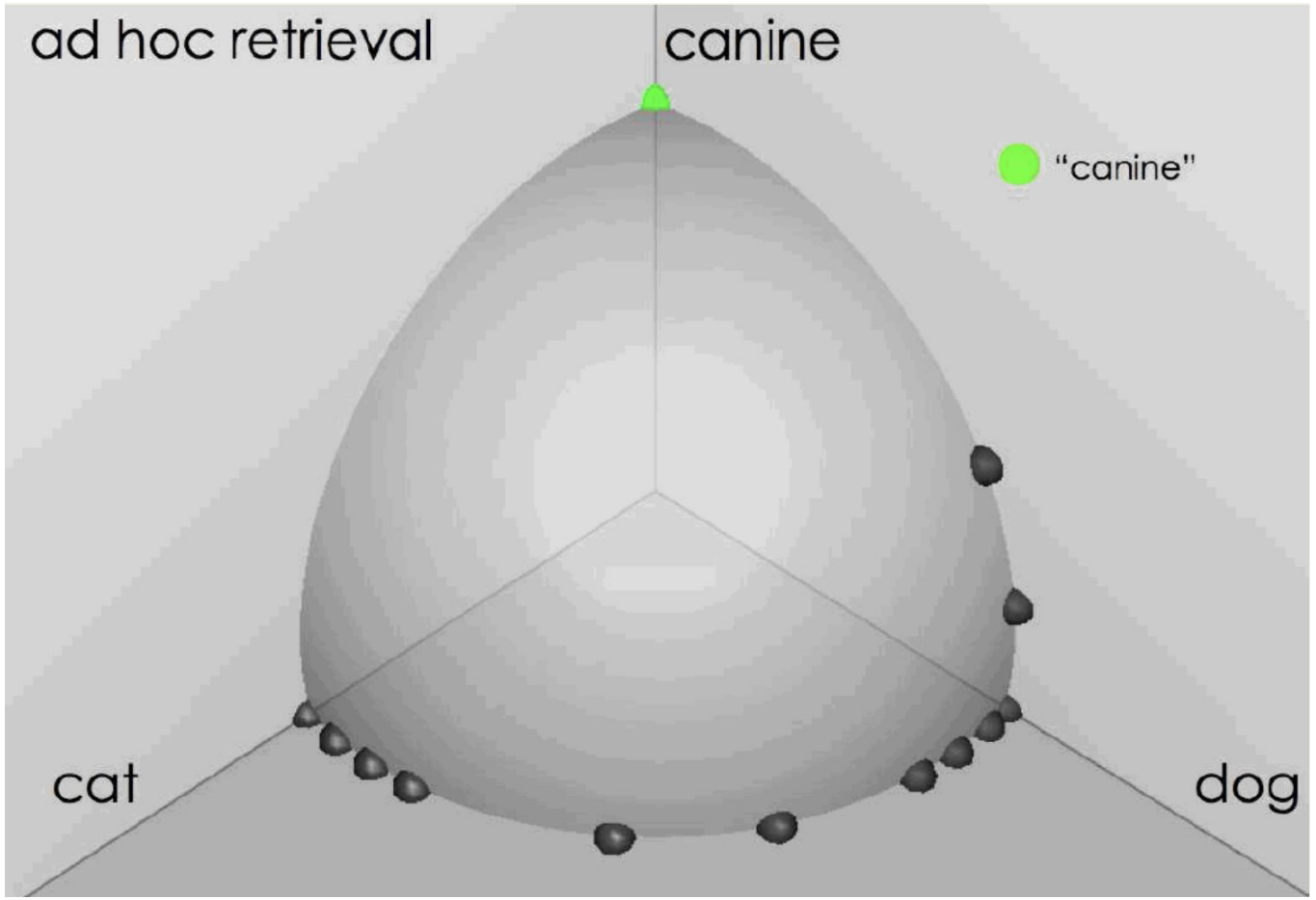
Relevance Feedback



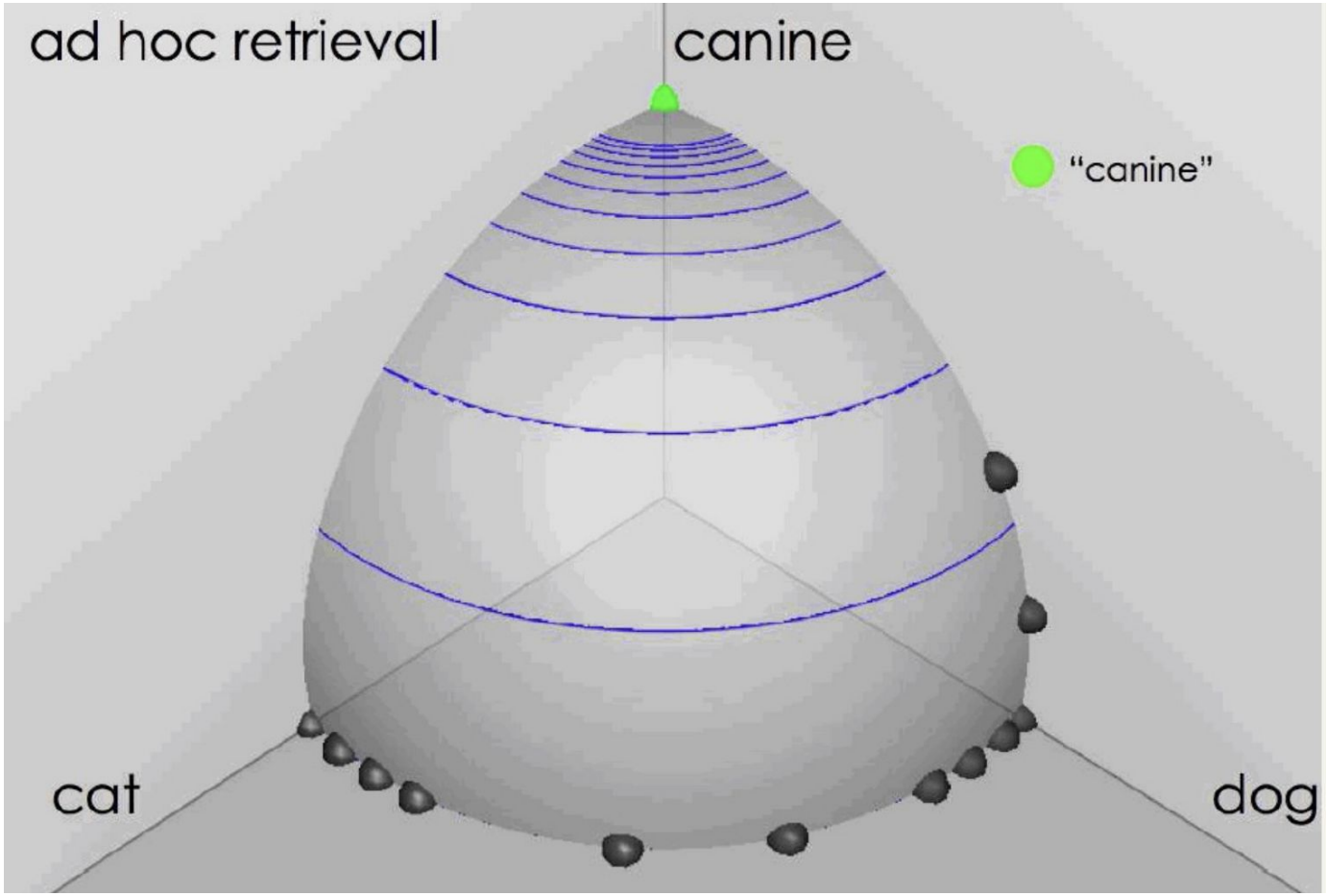
Results after Relevance Feedback



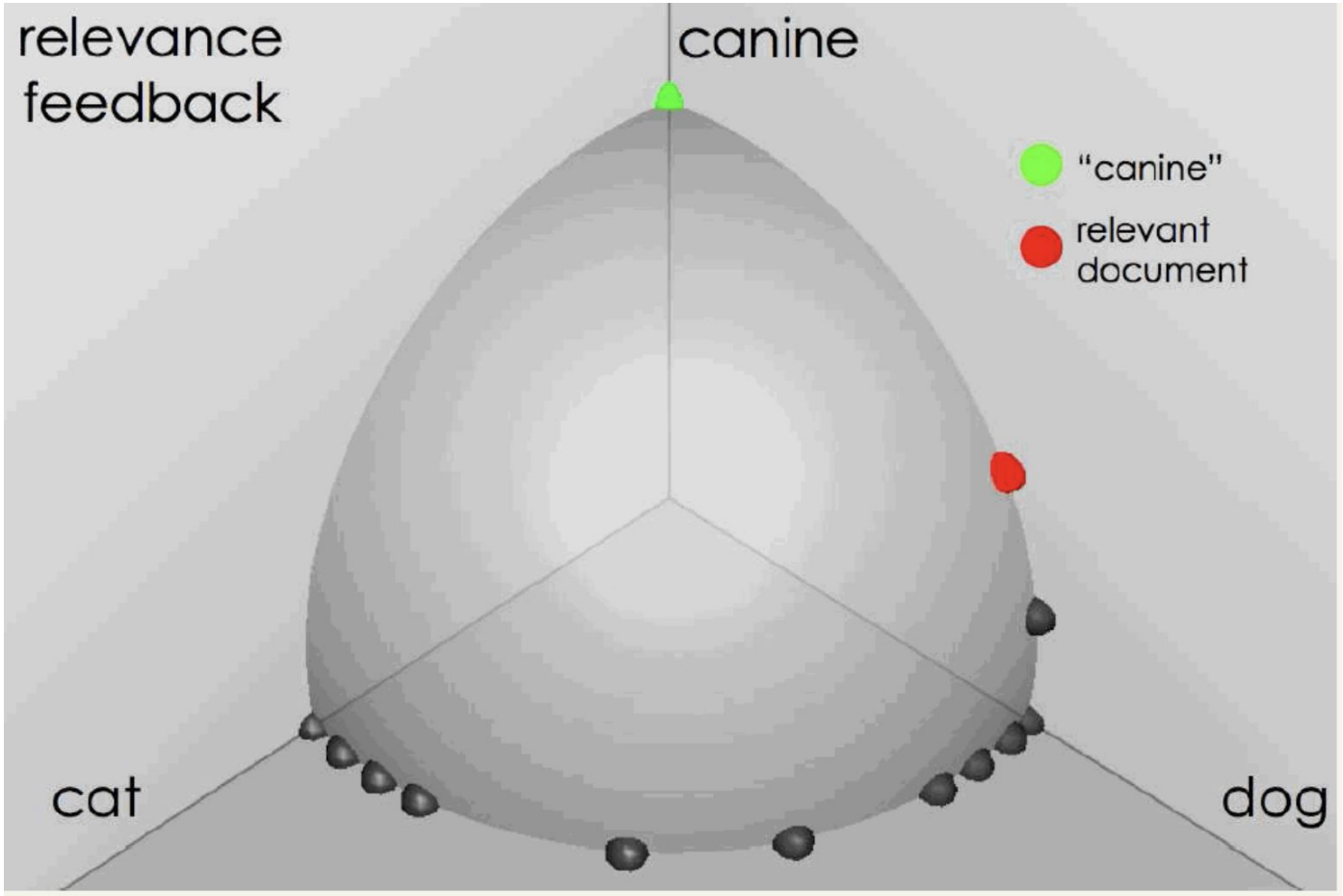
Ad hoc (for this) results for query canine



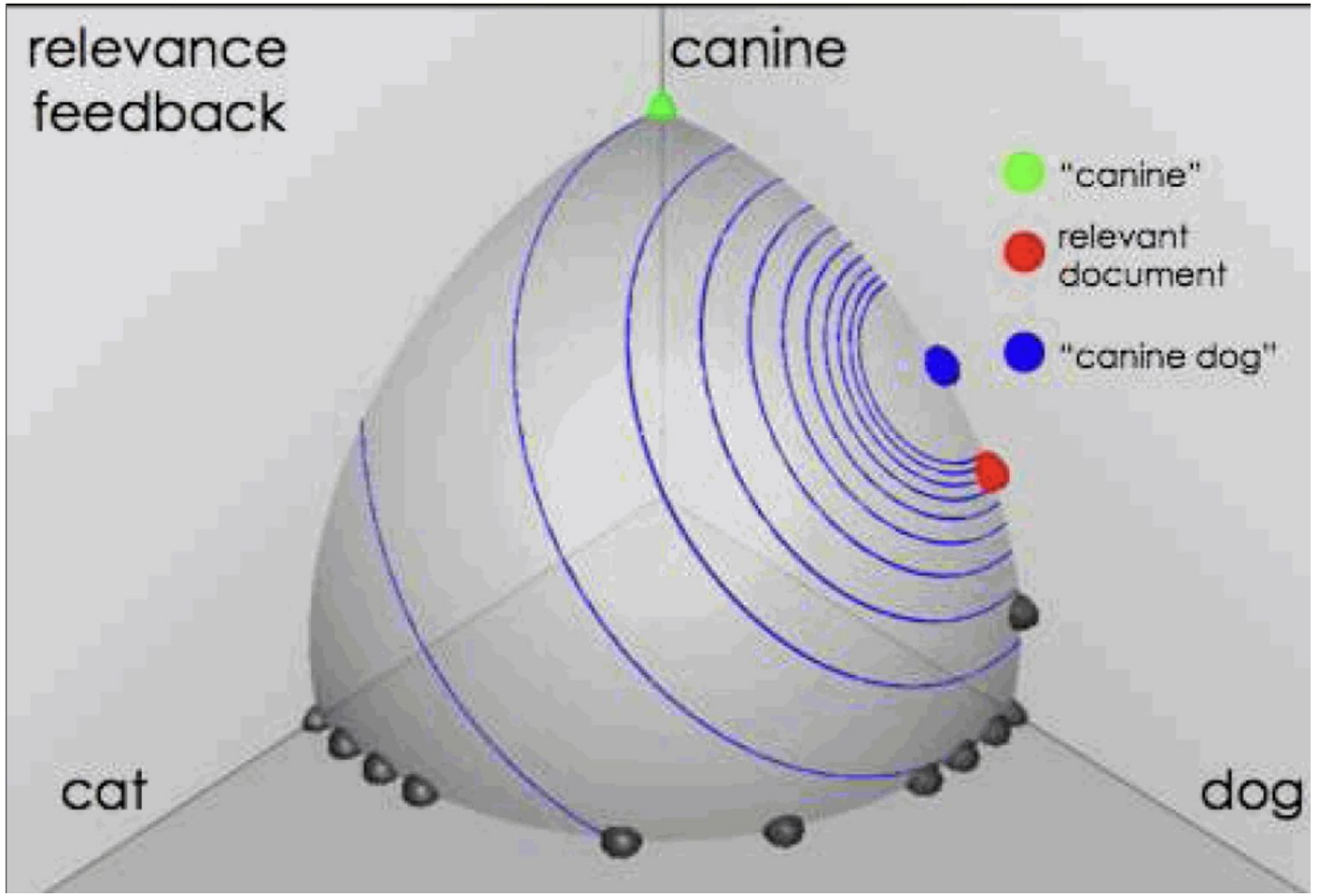
Ad hoc results for query canine



User feedback: Select what is relevant



Results after relevance feedback



Key concept: 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
- Definition: **Centroid**

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

where C is a set of documents.

Rocchio Algorithm

- The Rocchio algorithm uses the vector space model to pick a relevance feed-back query
- Rocchio seeks the query $\vec{q}_{opt}$ that maximizes

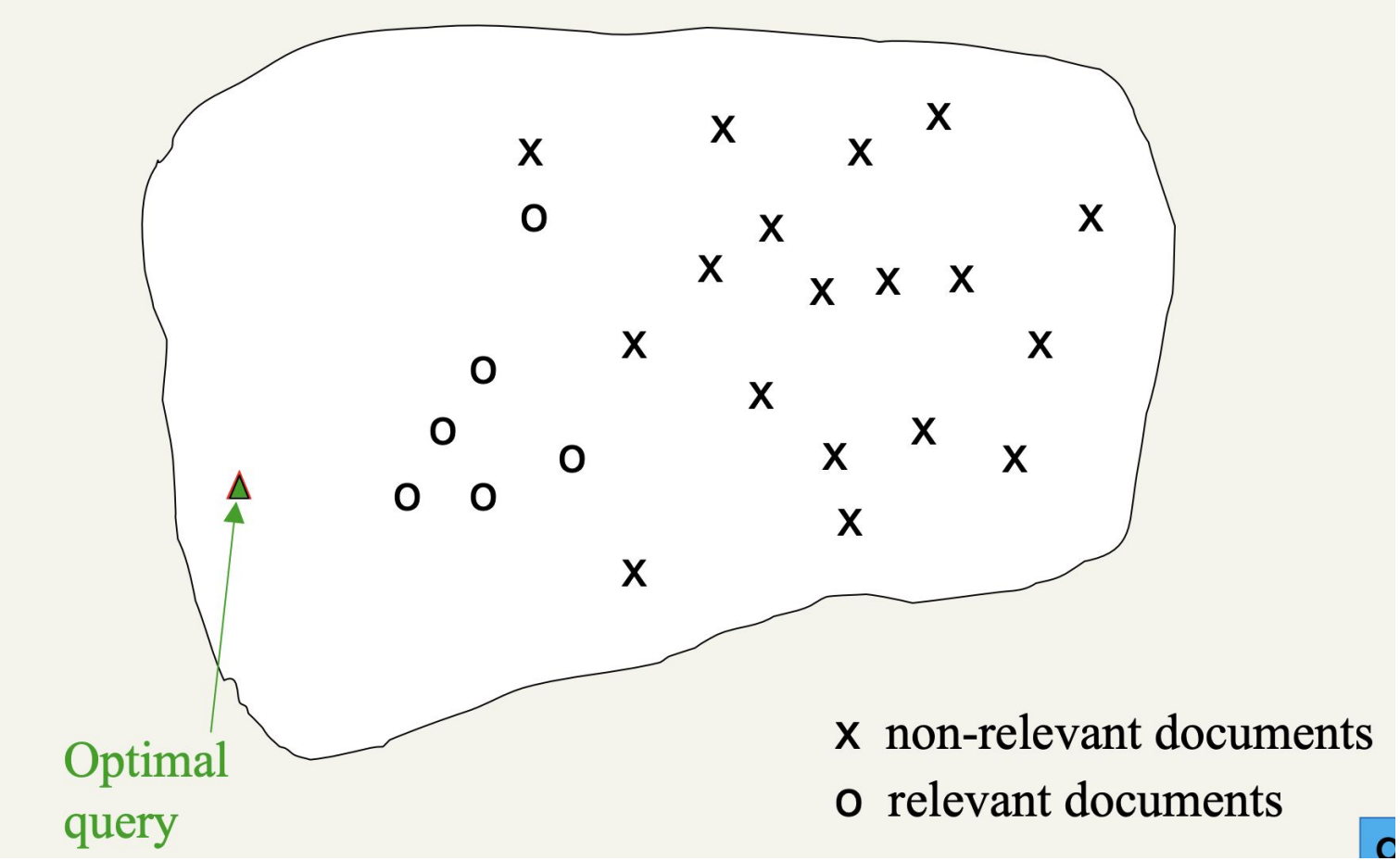
$$\vec{q}_{opt} = \arg\max_{\vec{q}} [\cos(\vec{q}, \vec{\mu}(C_r)) - \cos(\vec{q}, \vec{\mu}(C_{nr}))]$$

- Tries to separate docs marked relevant and non-relevant

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

- Problem: we don't know the truly relevant docs

The Theoretically Best Query



Rocchio 1971 Algorithm

- Used in practice:

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

- D_r = set of known relevant doc vectors
- D_{nr} = set of known irrelevant doc vectors
  - Different from C_r and C_{nr} !
- q_m = modified query vector; q_0 = original query vector; α,β,γ: weights (hand-chosen or set empirically)
- New query moves toward relevant documents and away from irrelevant documents

Subtleties to note

- Tradeoff α vs. β/γ : If we have a lot of judged documents, we want a higher β/γ.
- Some weights in query vector can go negative
  - Negative term weights are ignored (set to 0)

Query Modification: Example

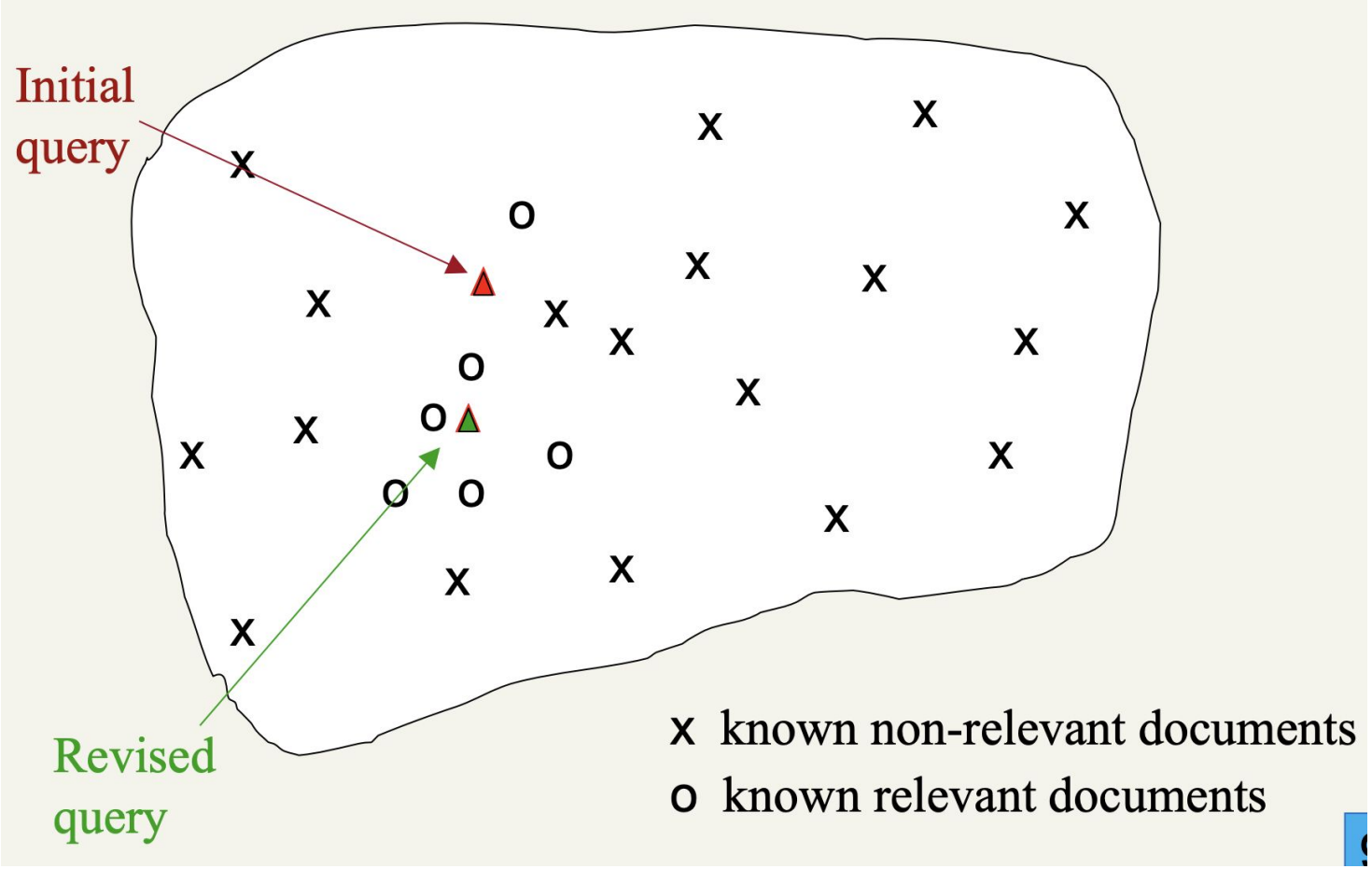
$$q = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \quad d_1 = \begin{pmatrix} 2 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad d_2 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 3 \\ 0 \end{pmatrix}, \quad d_3 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 2 \end{pmatrix}$$

$$\in R_+ \qquad \in R_+ \qquad \in R_-$$

$$\alpha = 0.5, \quad \beta = 0.25$$

$$q' = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} + 0.5 \left(\frac{1}{2} \begin{pmatrix} 2 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} + \frac{1}{2} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 3 \\ 0 \end{pmatrix} \right) - 0.25 \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 2 \end{pmatrix} = \begin{pmatrix} 0.75 \\ 1 \\ 0 \\ 1.75 \\ -0.5 \end{pmatrix}$$

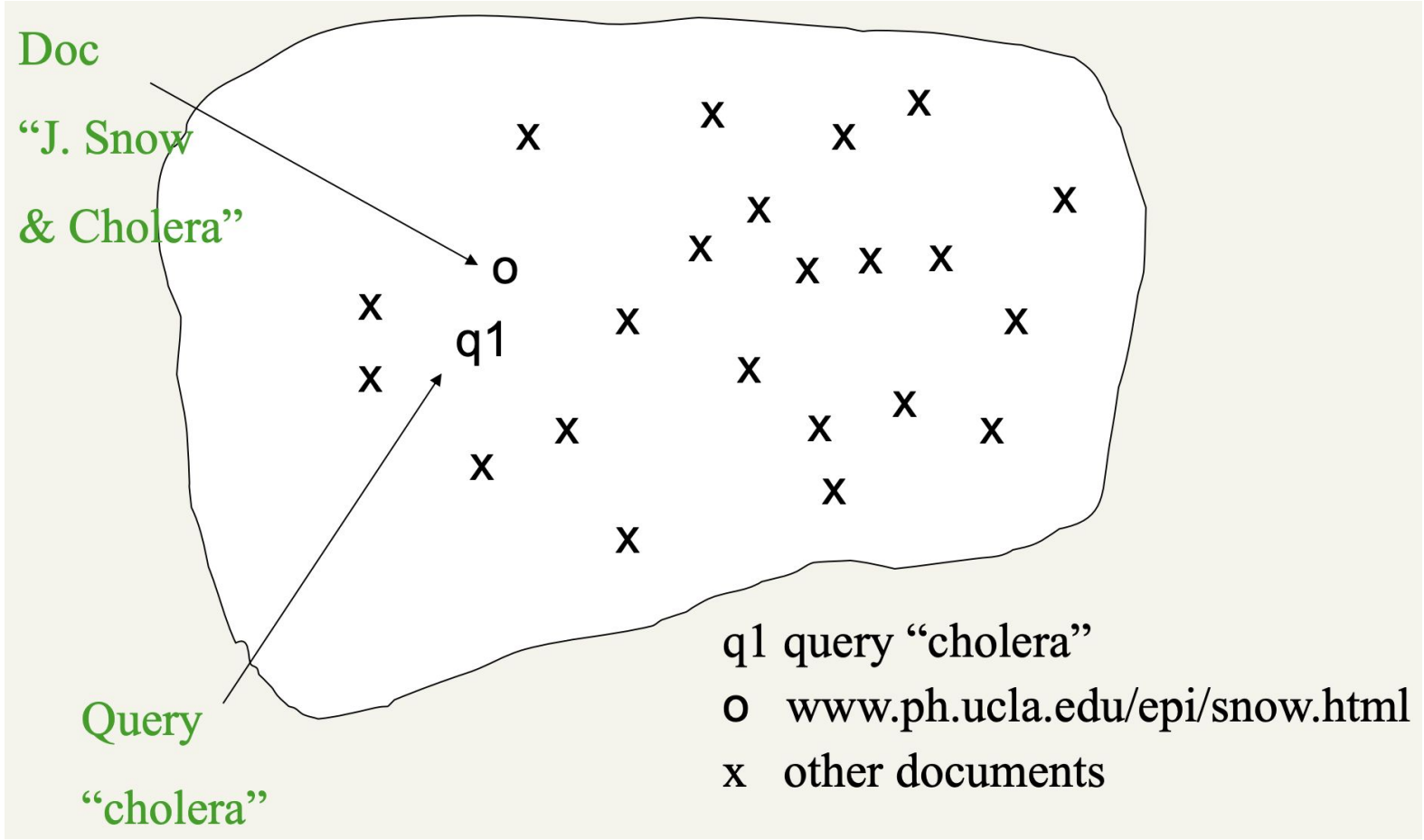
Relevance feedback on initial query



Relevance Feedback in vector spaces

- We can modify the query
- based on relevance feedback and apply standard vector space model.
- Use only the docs that were marked.
- Relevance feedback can improve recall and precision
- Relevance feedback is most useful for increasing recall in situations where recall is important
- Users can be expected to review results and to take time to iterate
- Positive feedback is more valuable than negative feedback (so, set $\beta > \gamma$; e.g. $\gamma = 0.25$, $\beta = 0.75$).

Aside: Vector Space can be Counterintuitive



High-dimensional Vector Spaces

- The queries “cholera” and “john snow” are far from each other in vector space.
- How can the document “John Snow and Cholera” be close to both of them?
- Our intuitions for 2- and 3-dimensional space don't work in >10,000 dimensions.
- 3 dimensions: If a document is close to many queries, then some of these queries must be close to each other.
- Doesn't hold for a high-dimensional space.

Relevance Feedback: Assumptions

- A1: User has sufficient knowledge for initial query.
- A2: Relevance prototypes are “well-behaved”.
  - Term distribution in relevant documents will be similar
  - Term distribution in non-relevant documents will be different from those in relevant documents
    - Either: All relevant documents are tightly clustered around a single prototype.
    - Or: There are different prototypes, but they have significant vocabulary overlap.
    - Similarities between relevant and irrelevant documents are small

Violation of A1

- User does not have sufficient initial knowledge.
- Examples:
- Misspellings (Tailar Suipht).
- Cross-language information retrieval (hígado).
- Mismatch of searcher’s vocabulary vs. collection vocabulary

Violation of A2

- Often: instances of a general concept, e.g. felines
- Good editorial content can address problem
  - Report on contradictory government policies

Relevance Feedback: Problems

- Long queries are inefficient for typical IR engine. Why?

Relevance Feedback: Problems

- Long queries are inefficient for typical IR engine.
  - Long response times for user.
  - High cost for retrieval system.
  - Partial solution:
    - Only reweight certain prominent terms
      - Perhaps top 20 by term frequency
- Users are often reluctant to provide explicit feedback
- It's often harder to understand why a particular document was retrieved after applying relevance feedback

Evaluation of relevance feedback strategies

- Use q_0 and compute precision and recall graph
- Use q_m and compute precision recall graph
  - Assess on all documents in the collection
    - Spectacular improvements, but … it's cheating!
    - Partly due to known relevant documents ranked higher
    - Must evaluate with respect to documents not seen by user
  - Use documents in residual collection (set of documents minus those assessed relevant)
    - Measures usually then lower than for original query
    - But a more realistic evaluation
    - Relative performance can be validly compared
- Empirically, one round of relevance feedback is often very useful. Two rounds is sometimes marginally useful.

Evaluation of relevance feedback strategies

- Second method – assess only the docs *not* rated by the user in the first round
  - Could make relevance feedback look worse than it really is
  - Can still assess relative performance of algorithms
- Most satisfactory – use two collections each with their own relevance assessments
  - q_0 and user feedback from first collection
  - q_m run on second collection and measured

Evaluation: Caveat

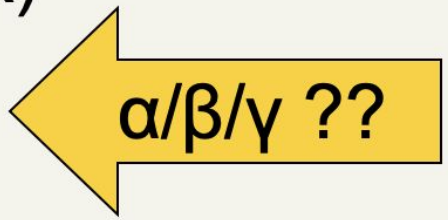
- True evaluation of usefulness must compare to other methods taking the same amount of time.
- Alternative to relevance feedback: User revises and resubmits query.
- Users may prefer revision/resubmission to having to judge relevance of documents.
- There is no clear evidence that relevance feedback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

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.
- Why?

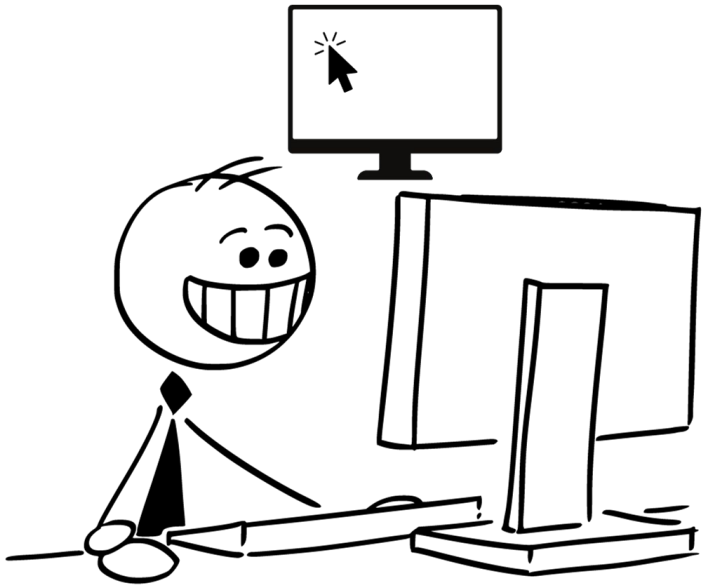
Relevance Feedback on the Web

- Some search engines offer a similar/related pages feature (this is a trivial form of relevance feedback)
  - Google (link-based)
  - Bing
  - Solr
- But some don't because it's hard to explain to average user:
  - msn live.com
  - Yahoo



Indirect relevance feedback

- On the web, DirectHit introduced a form of indirect relevance feedback.
- DirectHit ranked documents higher that users look at more often.
  - Clicked on links are assumed likely to be relevant
    - Assuming the displayed summaries are good, etc.
- Globally: Not necessarily user or query specific.
  - This is the general area of clickstream mining
- Today – handled as part of machine-learned ranking



Click-through Data as Implicit Data

- Idea: **Exploit clickthrough data** to optimize ranking functions
  - Advantage: does not add any overhead, does not put an additional burden on the user
- What is clickthrough data?
  - Formally a triplet: a **query**, a **ranking** presented to the user, and the **set of links** selected by the user
- Implicit relevance from clickthrough data
  - Selected links are interpreted as a (positive) implicit feedback
  - Assumes that users makes an **informed choice** (e.g. based on link text, text snippets or summaries)

Click-through Data: Example

1. **Kernel Machines**
   http://svm.first.gmd.de/
2. Support Vector Machine
   http://jbolivar.freeservers.com/
3. **SVM-Light Support Vector Machine**
   http://ais.gmd.de/ ~ thorsten/svm_light/
4. An Introduction to Support Vector Machines
   http://www.support – vector.net/
5. Support Vector Machine and Kernel Methods References
   http://svm.research.bell – labs.com/SVMrefs.html
6. Archives of SUPPORT-VECTOR-MACHINES@JISCMAIL.AC.UK
   http://www.jiscmail.ac.uk/lists/SUPPORT–VECTOR–MACHINES.html
7. **Lucent Technologies: SVM demo applet**
   http://svm.research.bell – labs.com/SVT/SVMsvt.html
8. Royal Holloway Support Vector Machine
   http://svm.dcs.rhbnc.ac.uk/
9. Support Vector Machine - The Software
   http://www.support – vector.net/software.html
10. Lagrangian Support Vector Machine Home Page
    http://www.cs.wisc.edu/dmi/lsvm

Ranking presented for the query "support vector machine".
Marked in bold are the links the user clicked on.

(taken from T. Joachims, 2002)

Making Sense of Click-through Data

- How can we make sense of clickthrough data?
  - Users are more likely to clique on a link if it is relevant to the query
  - But: the ranked list also introduces a **presentation bias**. Links further down the list are less likely to be even noticed than links higher up
  - Hypothesis: users click on the **relatively most promising** links among the top ones independent of their absolute relevance

- Extracting implicit data
  - **Pairwise preferences**: if a link l appearing after another link l' in a ranking is cliqued on, while l' is not cliqued on, then l is preferred over l'
  - Reasoning: l' has (probably) been noticed by the user (scanning order from top to bottom) once l is reached, so a decision has been made to clique on l but not on l'

Learning a Ranking Function

- Goal: How can one learn to better/optimally rank documents based on clickthrough data?
- Formal setting: define a family of ranking functions f and find an optimal f* within this family that respects the observed pairwise preferences
- More precisely with linear ranking functions:

$$f(d;q) = \langle \mathbf{w}, \Phi(d,q) \rangle, \quad where \ \ \mathbf{w}, \Phi(d,q) \in \Re^d$$

weights feature vector

Features Used for Ranking

What features should be used in the Φ function?

- Should describe the match between a document d and a query q
- Examples
  - number of words shared by query and document
  - number of shared words inside certain HTML tags
  - cosine similarity between query and document title or abstract
  - page rank of document d
  - rank of d in the result list of q for some search engine (e.g. within top10, within top50, etc.)
  - properties of the URL (contains tilde, length, etc.)
  - ...

Pairwise Preference Learning as Classification

This problem can be mapped to a classification problem as follows:

- **Convert** each pairwise preference into a binary classification instance by defining:

$$d \succ_q d' \mapsto (\Phi(d,q) - \Phi(d',q), +1) \text{ and } (\Phi(d',q) - \Phi(d,q), -1)$$

- Due to the linearity of the ranking function, one of these is sufficient, since

$$\langle \mathbf{w}, \Phi(d,q) - \Phi(d',q) \rangle = - \langle \mathbf{w}, \Phi(d',q) - \Phi(d,q) \rangle$$

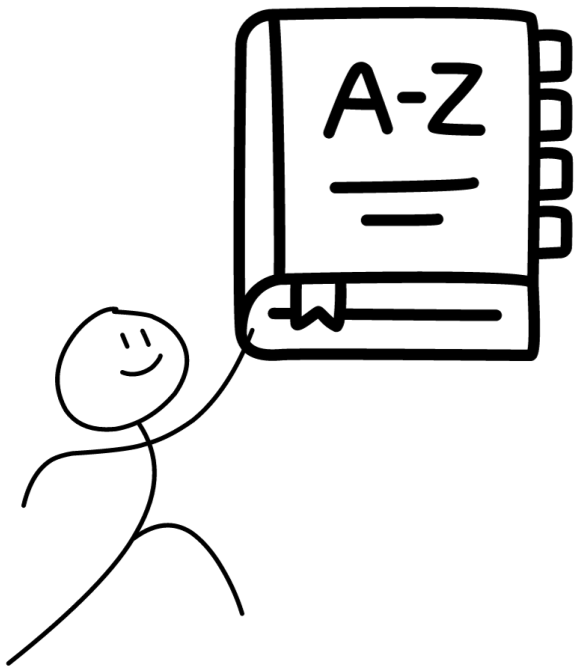
- Now we can use any classification problem by treating each obseved pairwise preference as an instance with binary label
- Can use known linear classification algorithms: e.g. perceptron or Maximum Margin Hyperplane (Support Vector Machine)

Query Expansion

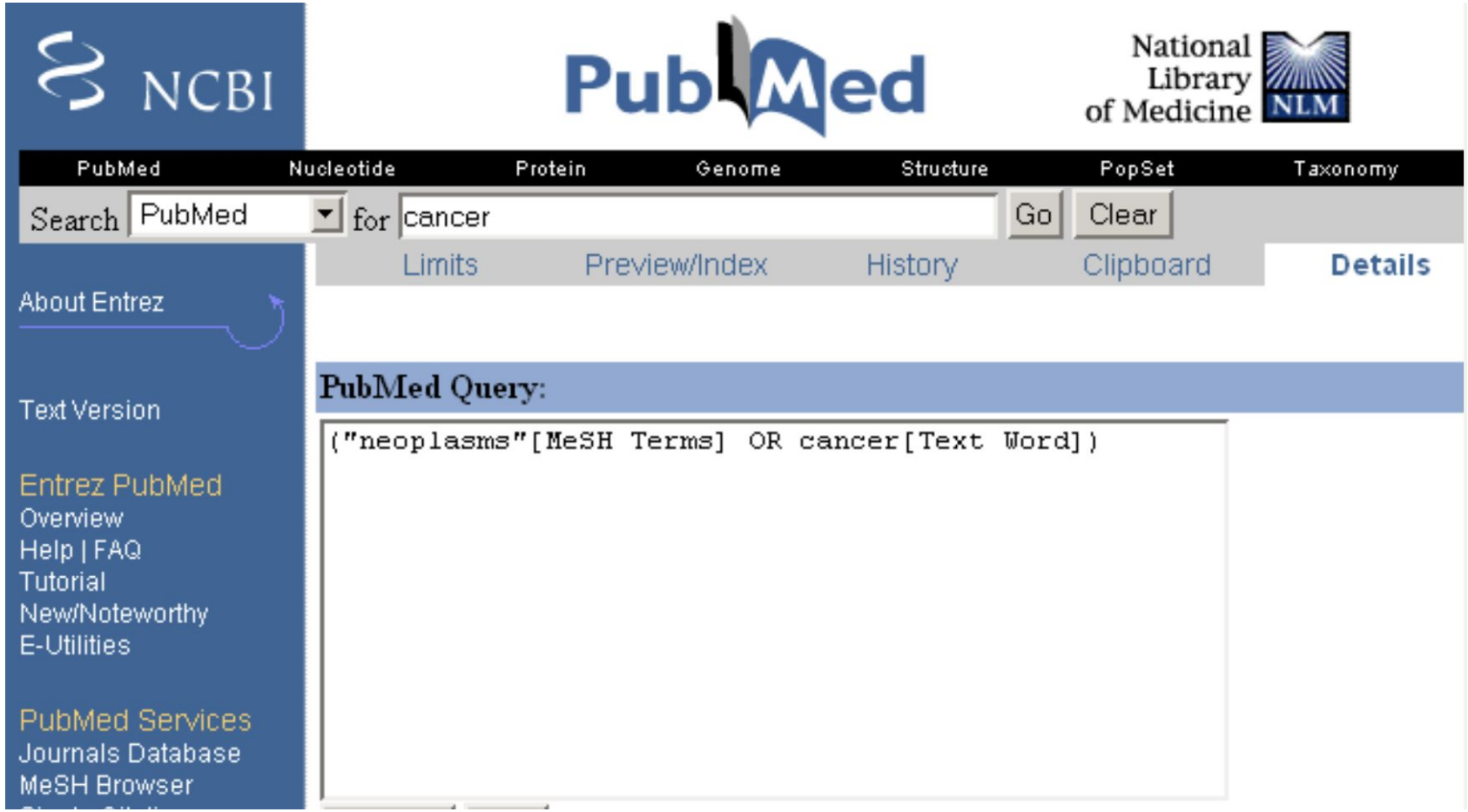
- In relevance feedback, users give additional input (relevant/non-relevant) on documents, which is used to reweight terms in the documents
- In query expansion, users give additional input (good/bad search term) on words or phrases

How do we augment the user query?

- Manual thesaurus
  - E.g. MedLine: physician, syn: doc, doctor, MD, medico
  - Can be query rather than just synonyms
- Global Analysis: (static; of all documents in collection)
  - Automatically derived thesaurus
    - (co-occurrence statistics)
  - Refinements based on query log mining
    - Common on the web
- Local Analysis: (dynamic)
  - Analysis of documents in result set



Example of manual thesaurus



Thesaurus-based query expansion

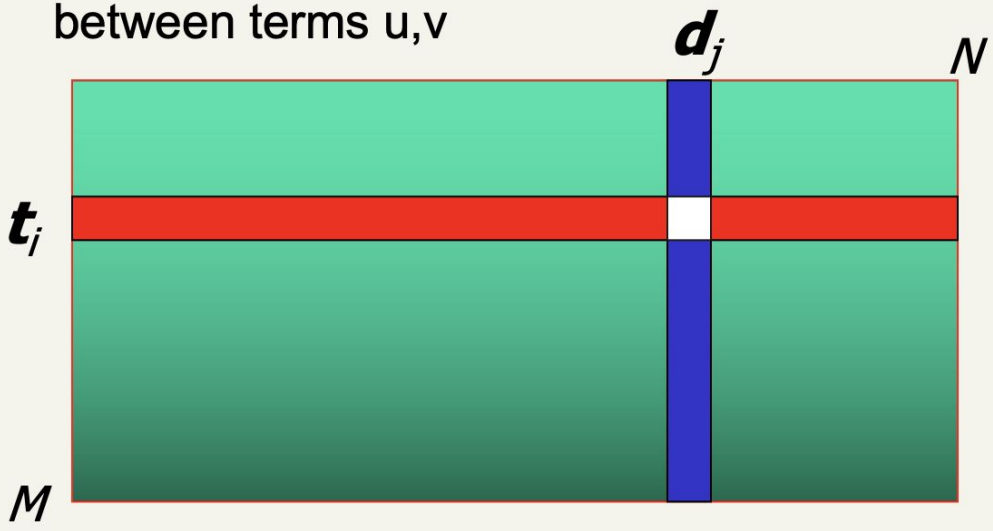
- For each term, *t*, in a query, expand the query with synonyms and related words of *t* from the thesaurus
  - feline → feline cat
- May weight added terms less than original query terms.
- Generally increases recall
- Widely used in many science/engineering fields
- May significantly decrease precision, particularly with ambiguous terms.
  - "interest rate" → "interest rate fascinate evaluate"
- There is a high cost of manually producing a thesaurus
  - And for updating it for scientific changes

Automatic Thesaurus Generation

- Attempt to generate a thesaurus automatically by analyzing the collection of documents
- Fundamental notion: similarity between two words
- Definition 1: Two words are similar if they co-occur with similar words.
- 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 Thesaurus

- Simplest way to compute one is based on term-term similarities in $C = AA^T$ where A is term-document matrix.
- $w_{i,j}$ = (normalized) weight for (t_i, d_j) $C_{u,v}$ is similarity score between terms u,v



What does C contain if A is a term-doc incidence (0/1) matrix?

- For each t_i, pick terms with high values in C

Automatic Thesaurus Generation Example

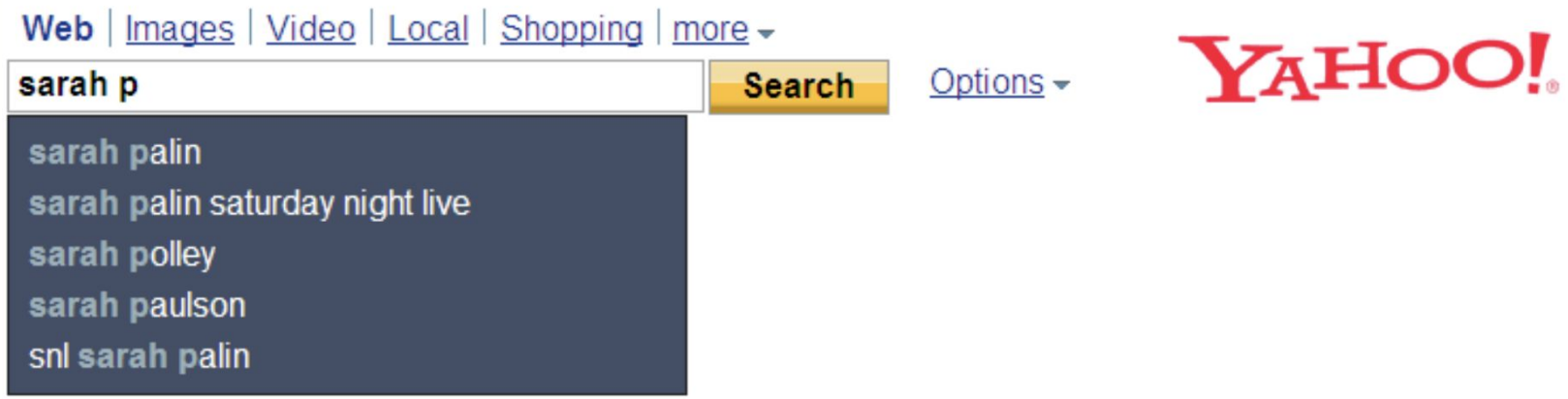
word	ten nearest neighbors
absolutely	absurd whatsoever totally exactly nothing
bottomed	dip copper drops topped slide trimmed slig
captivating	shimmer stunningly superbly plucky witty
doghouse	dog porch crawling beside downstairs gaze
Makeup	repellent lotion glossy sunscreen Skin gel p
mediating	reconciliation negotiate cease conciliation p
keeping	hoping bring wiping could some would oth
lithographs	drawings Picasso Dali sculptures Gauguin
pathogens	toxins bacteria organisms bacterial parasit
senses	grasp psyche truly clumsy naive innate aw

Automatic Thesaurus Generation Discussion

- Quality of associations is usually a problem.
- Term ambiguity may introduce irrelevant statistically correlated terms.
  - "Apple computer" → "Apple red fruit computer"
- Problems:
  - False positives: Words deemed similar that are not
  - False negatives: Words deemed dissimilar that are similar
- Since terms are highly correlated anyway, expansion may not retrieve many additional documents.

Query assist: Query Log Mining

- Generally done by query log mining
- Recommend frequent recent queries that contain partial string typed by user
- A ranking problem! View each prior query as a doc – Rank-order those matching partial string …



References

1. Slides provided by Sougata Saha (Instructor, Fall 2022 - CSE 4/535)
2. Materials provided by Dr. Rohini K Srihari
3. https://nlp.stanford.edu/IR-book/information-retrieval-book.html