9 *Relevance feedback and query expansion*

SYNONYMY

In most collections, the same concept may be referred to using different words. This issue, known as *synonymy*, has an impact on the recall of most information retrieval systems. For example, you would want a search for aircraft to match plane (but only for references to an *airplane*, not a woodworking plane), and for a search on thermodynamics to match references to heat in appropriate discussions. Users often attempt to address this problem themselves by manually refining a query, as was discussed in Section 1.4; in this chapter we discuss ways in which a system can help with query refinement, either fully automatically or with the user in the loop.

The methods for tackling this problem split into two major classes: global methods and local methods. Global methods are techniques for expanding or reformulating query terms independent of the query and results returned from it, so that changes in the query wording will cause the new query to match other semantically similar terms. Global methods include:

- Query expansion/reformulation with a thesaurus or WordNet (Section 9.2.2)
- Query expansion via automatic thesaurus generation (Section 9.2.3)
- Techniques like spelling correction (discussed in Chapter 3)

Local methods adjust a query relative to the documents that initially appear to match the query. The basic methods here are:

- Relevance feedback (Section 9.1)
- Pseudo relevance feedback, also known as Blind relevance feedback (Section 9.1.6)
- (Global) indirect relevance feedback (Section 9.1.7)

In this chapter, we will mention all of these approaches, but we will concentrate on relevance feedback, which is one of the most used and most successful approaches.

9.1 Relevance feedback and pseudo relevance feedback

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. In particular, the user gives feedback on the relevance of documents in an initial set of results. The basic procedure is:

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

Relevance feedback can go through one or more iterations of this sort. The process exploits the idea that it may be difficult to formulate a good query when you don't know the collection well, but it is easy to judge particular documents, and so it makes sense to engage in iterative query refinement of this sort. In such a scenario, relevance feedback can also be effective in tracking a user's evolving information need: seeing some documents may lead users to refine their understanding of the information they are seeking.

Image search provides a good example of relevance feedback. Not only is it easy to see the results at work, but this is a domain where a user can easily have difficulty formulating what they want in words, but can easily indicate relevant or nonrelevant images. After the user enters an initial query for bike on the demonstration system at:

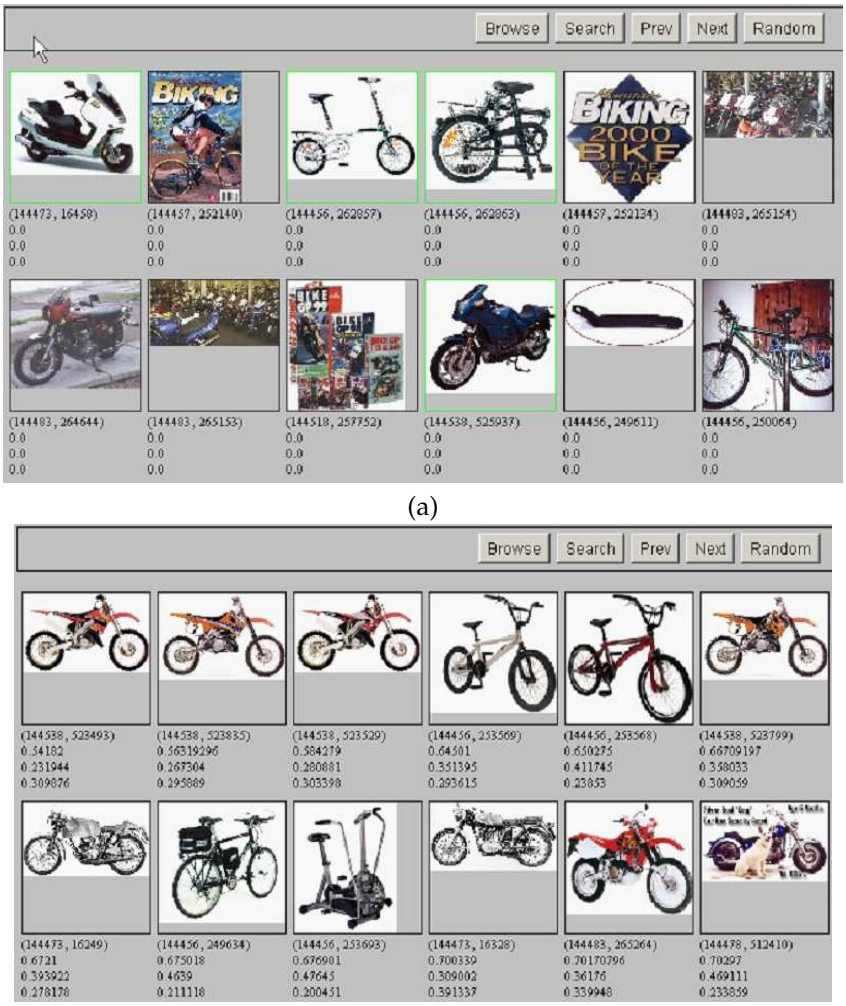
http://nayana.ece.ucsb.edu/imsearch/imsearch.html

the initial results (in this case, images) are returned. In Figure 9.1 (a), the user has selected some of them as relevant. These will be used to refine the query, while other displayed results have no effect on the reformulation. Figure 9.1 (b) then shows the new top-ranked results calculated after this round of relevance feedback.

Figure 9.2 shows a textual IR example where the user wishes to find out about new applications of space satellites.

9.1.1 The Rocchio algorithm for relevance feedback

The Rocchio Algorithm is the classic algorithm for implementing relevance feedback. It models a way of incorporating relevance feedback information into the vector space model of Section 6.3.



► **Figure 9.1** Relevance feedback searching over images. (a) The user views the initial query results for a query of bike, selects the first, third and fourth result in the top row and the fourth result in the bottom row as relevant, and submits this feedback. (b) The users sees the revised result set. Precision is greatly improved. From http://nayana.ece.ucsb.edu/imsearch/imsearch.html (Newsam et al. 2001).

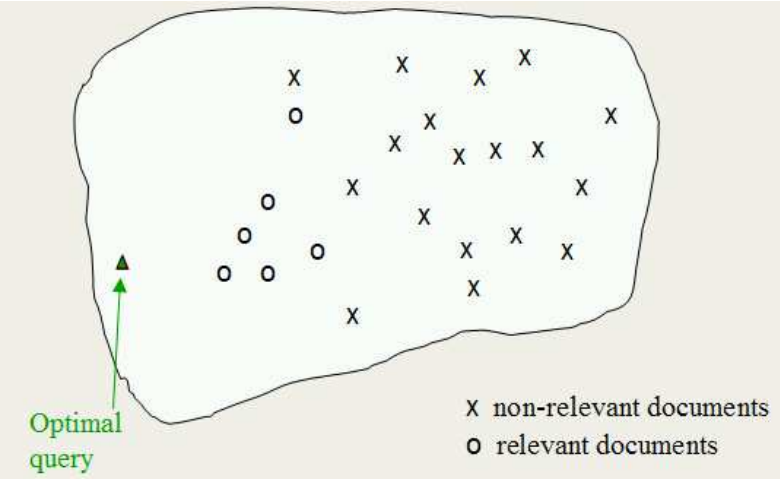
(a) Query: New space satellite applications

(b)
\+ 1. 0.539, 08/13/91, NASA Hasn't Scrapped Imaging Spectrometer
\+ 2. 0.533, 07/09/91, NASA Scratches Environment Gear From Satellite Plan
3. 0.528, 04/04/90, Science Panel Backs NASA Satellite Plan, But Urges Launches of Smaller Probes
4. 0.526, 09/09/91, A NASA Satellite Project Accomplishes Incredible Feat: Staying Within Budget
5. 0.525, 07/24/90, Scientist Who Exposed Global Warming Proposes Satellites for Climate Research
6. 0.524, 08/22/90, Report Provides Support for the Critics Of Using Big Satellites to Study Climate
7. 0.516, 04/13/87, Arianespace Receives Satellite Launch Pact From Telesat Canada
\+ 8. 0.509, 12/02/87, Telecommunications Tale of Two Companies

(c)
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

(d)
* 1. 0.513, 07/09/91, NASA Scratches Environment Gear From Satellite Plan
* 2. 0.500, 08/13/91, NASA Hasn't Scrapped Imaging Spectrometer
3. 0.493, 08/07/89, When the Pentagon Launches a Secret Satellite, Space Sleuths Do Some Spy Work of Their Own
4. 0.493, 07/31/89, NASA Uses 'Warm' Superconductors For Fast Circuit
* 5. 0.492, 12/02/87, Telecommunications Tale of Two Companies
6. 0.491, 07/09/91, Soviets May Adapt Parts of SS-20 Missile For Commercial Use
7. 0.490, 07/12/88, Gaping Gap: Pentagon Lags in Race To Match the Soviets In Rocket Launchers
8. 0.490, 06/14/90, Rescue of Satellite By Space Agency To Cost $90 Million

► **Figure 9.2** Example of relevance feedback on a text collection. (a) The initial query (a). (b) The user marks some relevant documents (shown with a plus sign). (c) The query is then expanded by 18 terms with weights as shown. (d) The revised top results are then shown. A * marks the documents which were judged relevant in the relevance feedback phase.



► **Figure 9.3** The Rocchio optimal query for separating relevant and nonrelevant documents.

The underlying theory. We want to find a query vector, denoted as $\vec{q}$, that maximizes similarity with relevant documents while minimizing similarity with nonrelevant documents. If C_r is the set of relevant documents and C_{nr} is the set of nonrelevant documents, then we wish to find:[1]

$$\vec{q}_{opt} = \arg\max_{\vec{q}} [\text{sim}(\vec{q}, C_r) - \text{sim}(\vec{q}, C_{nr})], \tag{9.1}$$

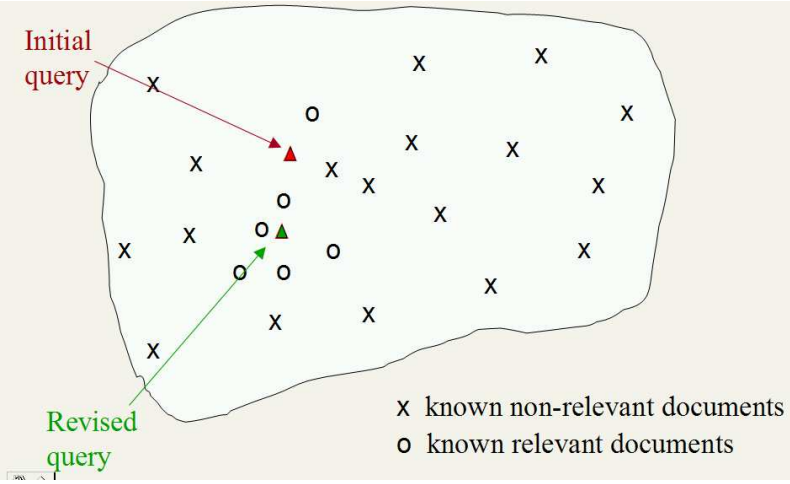
where sim is defined as in Equation 6.10. Under cosine similarity, the optimal query vector $\vec{q}_{opt}$ for separating the relevant and nonrelevant documents is:

$$\vec{q}_{opt} = \frac{1}{|C_r|} \sum_{\vec{d}_j \in C_r} \vec{d}_j - \frac{1}{|C_{nr}|} \sum_{\vec{d}_j \in C_{nr}} \vec{d}_j \tag{9.2}$$

That is, the optimal query is the vector difference between the centroids of the relevant and nonrelevant documents; see Figure 9.3. However, this observation is not terribly useful, precisely because the full set of relevant documents is not known: it is what we want to find.

ROCCHIO ALGORITHM **The Rocchio (1971) algorithm.** This was the relevance feedback mecha-

1. In the equation, $\arg\max_x f(x)$ returns a value of x which maximizes the value of the function $f(x)$. Similarly, $\arg\min_x f(x)$ returns a value of x which minimizes the value of the function $f(x)$.



► **Figure 9.4** An application of Rocchio's algorithm. Some documents have been labeled as relevant and nonrelevant and the initial query vector is moved in response to this feedback.

nism introduced in and popularized by Salton's SMART system around 1970. In a real IR query context, we have a user query and partial knowledge of known relevant and nonrelevant documents. The algorithm proposes using the modified query $\vec{q}_m$:

$$\vec{q}_m = \alpha \vec{q}_0 + \beta \frac{1}{|D_r|} \sum_{\vec{d}_j \in D_r} \vec{d}_j - \gamma \frac{1}{|D_{nr}|} \sum_{\vec{d}_j \in D_{nr}} \vec{d}_j \tag{9.3}$$

where q_0 is the original query vector, D_r and D_{nr} are the set of known relevant and nonrelevant documents respectively, and α, β, and γ are weights attached to each term. These control the balance between trusting the judged document set versus the query: if we have a lot of judged documents, we would like a higher β and γ. Starting from q_0, the new query moves you some distance toward the centroid of the relevant documents and some distance away from the centroid of the nonrelevant documents. This new query can be used for retrieval in the standard vector space model (see Section 6.3). We can easily leave the positive quadrant of the vector space by subtracting off a nonrelevant document's vector. In the Rocchio algorithm, negative term weights are ignored. That is, the term weight is set to 0. Figure 9.4 shows the effect of applying relevance feedback.

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. Positive feedback also turns out to be much more valuable than negative feedback, and so most IR systems set $\gamma < \beta$. Reasonable values might be $\alpha = 1$, $\beta = 0.75$, and $\gamma = 0.15$. In fact, many systems, such as the image search system in Figure 9.1, allow only positive feedback, which is equivalent to setting $\gamma = 0$. Another alternative is to use only the marked nonrelevant document which received the highest ranking from the IR system as negative feedback (here, $|D_{nr}| = 1$ in Equation (9.3)). While many of the experimental results comparing various relevance feedback variants are rather inconclusive, some studies have suggested that this variant, called *Ide dec-hi* is the most effective or at least the most consistent performer.

IDE DEC-HI

9.1.2 Probabilistic relevance feedback

Rather than reweighting the query in a vector space, if a user has told us some relevant and nonrelevant documents, then we can proceed to build a classifier. One way of doing this is with a Naive Bayes probabilistic model. If R is a Boolean indicator variable expressing the relevance of a document, then we can estimate $P(x_t = 1|R)$, the probability of a term t appearing in a document, depending on whether it is relevant or not, as:

$$\begin{aligned} \hat{P}(x_t = 1|R = 1) &= |VR_t|/|VR| \\ \hat{P}(x_t = 1|R = 0) &= (df_t - |VR_t|)/(N - |VR|) \end{aligned} \tag{9.4}$$

where N is the total number of documents, df_t is the number that contain t, VR is the set of known relevant documents, and VR_t is the subset of this set containing t. Even though the set of known relevant documents is a perhaps small subset of the true set of relevant documents, if we assume that the set of relevant documents is a small subset of the set of all documents then the estimates given above will be reasonable. This gives a basis for another way of changing the query term weights. We will discuss such probabilistic approaches more in Chapters 11 and 13, and in particular outline the application to relevance feedback in Section 11.3.4 (page 228). For the moment, observe that using just Equation (9.4) as a basis for term-weighting is likely insufficient. The equations use only collection statistics and information about the term distribution within the documents judged relevant. They preserve no memory of the original query.

9.1.3 When does relevance feedback work?

The success of relevance feedback depends on certain assumptions. Firstly, the user has to have sufficient knowledge to be able to make an initial query

which is at least somewhere close to the documents they desire. This is needed anyhow for successful information retrieval in the basic case, but it is important to see the kinds of problems that relevance feedback cannot solve alone. Cases where relevance feedback alone is not sufficient include:

- Misspellings. If the user spells a term in a different way to the way it is spelled in any document in the collection, then relevance feedback is unlikely to be effective. This can be addressed by the spelling correction techniques of Chapter 3.
- Cross-language information retrieval. Documents in another language are not nearby in a vector space based on term distribution. Rather, documents in the same language cluster more closely together.
- Mismatch of searcher's vocabulary versus collection vocabulary. If the user searches for laptop but all the documents use the term notebook computer, then the query will fail, and relevance feedback is again most likely ineffective.

Secondly, the relevance feedback approach requires relevant documents to be similar to each other. That is, they should cluster. Ideally, the term distribution in all relevant documents will be similar to that in the documents marked by the users, while the term distribution in all nonrelevant documents will be different from those in relevant documents. Things will work well if all relevant documents are tightly clustered around a single prototype, or, at least, if there are different prototypes, if the relevant documents have significant vocabulary overlap, while similarities between relevant and nonrelevant documents are small. Implicitly, the Rocchio relevance feedback model treats relevant documents as a single *cluster*, which it models via the centroid of the cluster. This approach does not work as well if the relevant documents are a multimodal class, that is, they consist of several clusters of documents within the vector space. This can happen with:

- Subsets of the documents using different vocabulary, such as Burma vs. Myanmar
- A query for which the answer set is inherently disjunctive, such as Pop stars who once worked at Burger King.
- Instances of a general concept, which often appear as a disjunction of more specific concepts, for example, felines.

Good editorial content in the collection can often provide a solution to this problem. For example, an article on the attitudes of different groups to the situation in Burma could introduce the terminology used by different parties, thus linking the document clusters.

Relevance feedback is not necessarily popular with users. Users are often reluctant to provide explicit feedback, or in general do not wish to prolong the search interaction. Furthermore, it is often harder to understand why a particular document was retrieved after relevance feedback is applied.

Relevance feedback can also have practical problems. The long queries that are generated by straightforward application of relevance feedback techniques are inefficient for a typical IR system. This results in a high computing cost for the retrieval and potentially long response times for the user. A partial solution to this is to only reweight certain prominent terms in the relevant documents, such as perhaps the top 20 terms by term frequency. Some experimental results have also suggested that using a limited number of terms like this may give better results (Harman 1992) though other work has suggested that using more terms is better in terms of retrieved document quality (Buckley et al. 1994b).

9.1.4 Relevance feedback on the web

Some web search engines offer a similar/related pages feature: the user indicates a document in the results set as exemplary from the standpoint of meeting his information need and requests more documents like it. This can be viewed as a particular simple form of relevance feedback. However, in general relevance feedback has been little used in web search. One exception was the Excite web search engine, which initially provided full relevance feedback. However, the feature was in time dropped, due to lack of use. On the web, few people use advanced search interfaces and most would like to complete their search in a single interaction. But the lack of uptake also probably reflects two other factors: relevance feedback is hard to explain to the average user, and relevance feedback is mainly a recall enhancing strategy, and web search users are only rarely concerned with getting sufficient recall.

Spink et al. (2000) present results from the use of relevance feedback in the Excite search engine. Only about 4% of user query sessions used the relevance feedback option, and these were usually exploiting the "More like this" link next to each result. About 70% of users only looked at the first page of results and did not pursue things any further. For people who used relevance feedback, results were improved about two thirds of the time.

An important more recent thread of work is the use of clickstream data (what links a user clicks on) to provide indirect relevance feedback. Use of this data is studied in detail in (Joachims 2002b, Joachims et al. 2005). The very successful use of web link structure (see Chapter 21) can also be viewed as implicit feedback, but provided by page authors rather than readers (though in practice most authors are also readers).

? **Exercise 9.1**

In Rocchio's algorithm, what weight setting for $\alpha/\beta/\gamma$ does a "Find pages like this one" search correspond to?

Exercise 9.2 [⋆]

Give three reasons why relevance feedback has been little used in web search.

9.1.5 Evaluation of relevance feedback strategies

Interactive relevance feedback can give very substantial gains in retrieval performance. Empirically, one round of relevance feedback is often very useful. Two rounds is sometimes marginally more useful. Successful use of relevance feedback requires enough judged documents, otherwise the process is unstable in that it may drift away from the user's information need. Accordingly, having at least five judged documents is recommended.

There is some subtlety to evaluating the effectiveness of relevance feedback in a sound and enlightening way. The obvious first strategy is to start with an initial query q_0 and to compute a precision-recall graph. Following one round of feedback from the user, we compute the modified query q_m and again compute a precision-recall graph. Here, in both rounds we assess performance over all documents in the collection, which makes comparisons straightforward. If we do this, we find spectacular gains from relevance feedback: gains on the order of 50% in mean average precision. But unfortunately it is cheating. The gains are partly due to the fact that known relevant documents (judged by the user) are now ranked higher. Fairness demands that we should only evaluate with respect to documents not seen by the user.

A second idea is to use documents in the *residual collection* (the set of documents minus those assessed relevant) for the second round of evaluation. This seems like a more realistic evaluation. Unfortunately, the measured performance can then often be lower than for the original query. This is particularly the case if there are few relevant documents, and so a fair proportion of them have been judged by the user in the first round. The relative performance of variant relevance feedback methods can be validly compared, but it is difficult to validly compare performance with and without relevance feedback because the collection size and the number of relevant documents changes from before the feedback to after it.

Thus neither of these methods is fully satisfactory. A third method is to have two collections, one which is used for the initial query and relevance judgments, and the second that is then used for comparative evaluation. The performance of both q_0 and q_m can be validly compared on the second collection.

Perhaps the best evaluation of the utility of relevance feedback is to do user studies of its effectiveness, in particular by doing a time-based comparison:

	Precision at $k = 50$	
Term weighting	no RF	pseudo RF
lnc.ltc	64.2%	72.7%
Lnu.ltu	74.2%	87.0%

► **Figure 9.5** Results showing pseudo relevance feedback greatly improving performance. These results are taken from the Cornell SMART system at TREC 4 (Buckley et al. 1995), and also contrast the use of two different length normalization schemes (L vs. l); cf. Figure 6.15 (page 128). Pseudo relevance feedback consisted of adding 20 terms to each query.

how fast does a user find relevant documents with relevance feedback vs. another strategy (such as query reformulation), or alternatively, how many relevant documents does a user find in a certain amount of time. Such notions of user utility are fairest and closest to real system usage.

9.1.6 Pseudo relevance feedback

PSEUDO RELEVANCE FEEDBACK
BLIND RELEVANCE FEEDBACK

Pseudo relevance feedback, also known as *blind relevance feedback*, provides a method for automatic local analysis. It automates the manual part of relevance feedback, so that the user gets improved retrieval performance without an extended interaction. The method is to do normal retrieval to find an initial set of most relevant documents, to then *assume* that the top k ranked documents are relevant, and finally to do relevance feedback as before under this assumption.

This automatic technique mostly works. Evidence suggests that it tends to work better than global analysis (Section 9.2). It has been found to improve performance in the TREC ad hoc task. See for example the results in Figure 9.5. But it is not without the dangers of an automatic process. For example, if the query is about copper mines and the top several documents are all about mines in Chile, then there may be query drift in the direction of documents on Chile.

9.1.7 Indirect relevance feedback

IMPLICIT RELEVANCE FEEDBACK

We can also use indirect sources of evidence rather than explicit feedback on relevance as the basis for relevance feedback. This is often called *implicit (relevance) feedback*. Implicit feedback is less reliable than explicit feedback, but is more useful than pseudo relevance feedback, which contains no evidence of user judgments. Moreover, while users are often reluctant to provide explicit feedback, it is easy to collect implicit feedback in large quantities for a high volume system, such as a web search engine.

On the web, DirectHit introduced the idea of ranking more highly documents that users chose to look at more often. In other words, clicks on links were assumed to indicate that the page was likely relevant to the query. This approach makes various assumptions, such as that the document summaries displayed in results lists (on whose basis users choose which documents to click on) are indicative of the relevance of these documents. In the original DirectHit search engine, the data about the click rates on pages was gathered globally, rather than being user or query specific. This is one form of the general area of *clickstream mining*. Today, a closely related approach is used in ranking the advertisements that match a web search query (Chapter 19).

CLICKSTREAM MINING

9.1.8 Summary

Relevance feedback has been shown to be very effective at improving relevance of results. Its successful use requires queries for which the set of relevant documents is medium to large. Full relevance feedback is often onerous for the user, and its implementation is not very efficient in most IR systems. In many cases, other types of interactive retrieval may improve relevance by about as much with less work.

Beyond the core ad hoc retrieval scenario, other uses of relevance feedback include:

- Following a changing information need (e.g., names of car models of interest change over time)
- Maintaining an information filter (e.g., for a news feed). Such filters are discussed further in Chapter 13.
- Active learning (deciding which examples it is most useful to know the class of to reduce annotation costs).

?

Exercise 9.3

Under what conditions would the modified query q_m in Equation 9.3 be the same as the original query q_0? In all other cases, is q_m closer than q_0 to the centroid of the relevant documents?

Exercise 9.4

Why is positive feedback likely to be more useful than negative feedback to an IR system? Why might only using one nonrelevant document be more effective than using several?

Exercise 9.5

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 as in Equation (9.3) what would the revised query vector be after relevance feedback? Assume $\alpha = 1, \beta = 0.75, \gamma = 0.25$.

Exercise 9.6 [⋆]

Omar has implemented a relevance feedback web search system, where he is going to do relevance feedback based only on words in the title text returned for a page (for efficiency). The user is going to rank 3 results. The first user, Jinxing, queries for:

banana slug

and the top three titles returned are:

banana slug Ariolimax columbianus
Santa Cruz mountains banana slug
Santa Cruz Campus Mascot

Jinxing judges the first two documents relevant, and the third nonrelevant. Assume that Omar's search engine uses term frequency but no length normalization nor IDF. Assume that he is using the Rocchio relevance feedback mechanism, with $\alpha = \beta = \gamma = 1$. Show the final revised query that would be run. (Please list the vector elements in alphabetical order.)

9.2 Global methods for query reformulation

In this section we more briefly discuss three global methods for expanding a query: by simply aiding the user in doing so, by using a manual thesaurus, and through building a thesaurus automatically.

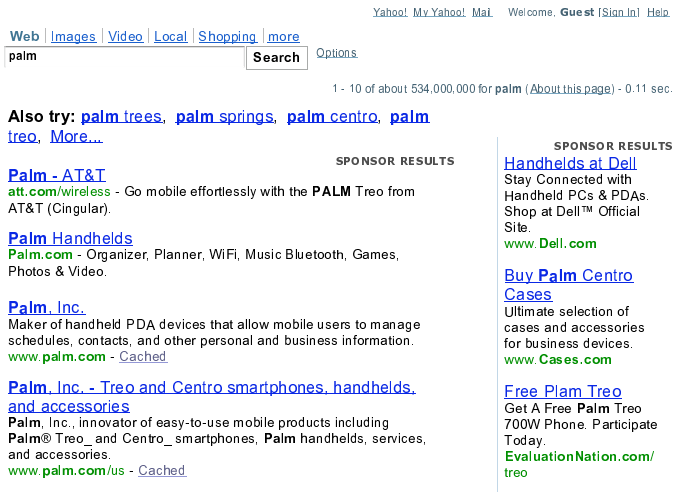
9.2.1 Vocabulary tools for query reformulation

Various user supports in the search process can help the user see how their searches are or are not working. This includes information about words that were omitted from the query because they were on stop lists, what words were stemmed to, the number of hits on each term or phrase, and whether words were dynamically turned into phrases. The IR system might also suggest search terms by means of a thesaurus or a controlled vocabulary. A user can also be allowed to browse lists of the terms that are in the inverted index, and thus find good terms that appear in the collection.

9.2.2 Query expansion

In relevance feedback, users give additional input on documents (by marking documents in the results set as relevant or not), and this input is used to reweight the terms in the query for documents. In *query expansion* on the other hand, users give additional input on query words or phrases, possibly suggesting additional query terms. Some search engines (especially on the

QUERY EXPANSION



► **Figure 9.6** An example of query expansion in the interface of the Yahoo! web search engine in 2006. The expanded query suggestions appear just below the "Search Results" bar.

web) suggest related queries in response to a query; the users then opt to use one of these alternative query suggestions. Figure 9.6 shows an example of query suggestion options being presented in the Yahoo! web search engine. The central question in this form of query expansion is how to generate alternative or expanded queries for the user. 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. Use of a thesaurus can be combined with ideas of term weighting: for instance, one might weight added terms less than original query terms.

Methods for building a thesaurus for query expansion include:

- Use of a controlled vocabulary that is maintained by human editors. Here, there is a canonical term for each concept. The subject headings of traditional library subject indexes, such as the Library of Congress Subject Headings, or the Dewey Decimal system are examples of a controlled vocabulary. Use of a controlled vocabulary is quite common for well-resourced domains. A well-known example is the Unified Medical Language System (UMLS) used with MedLine for querying the biomedical research literature. For example, in Figure 9.7, neoplasms was added to a

- User query: `cancer`
- PubMed query: ("neoplasms"[TIAB] NOT Medline[SB]) OR "neoplasms"[MeSH Terms] OR cancer[Text Word]
- User query: `skin itch`
- PubMed query: ("skin"[MeSH Terms] OR ("integumentary system"[TIAB] NOT Medline[SB]) OR "integumentary system"[MeSH Terms] OR skin[Text Word]) AND (("pruritus"[TIAB] NOT Medline[SB]) OR "pruritus"[MeSH Terms] OR itch[Text Word])

► **Figure 9.7** Examples of query expansion via the PubMed thesaurus. When a user issues a query on the PubMed interface to Medline at http://www.ncbi.nlm.nih.gov/entrez/, their query is mapped on to the Medline vocabulary as shown.

search for cancer. This Medline query expansion also contrasts with the Yahoo! example. The Yahoo! interface is a case of interactive query expansion, whereas PubMed does automatic query expansion. Unless the user chooses to examine the submitted query, they may not even realize that query expansion has occurred.

- A manual thesaurus. Here, human editors have built up sets of synonymous names for concepts, without designating a canonical term. The UMLS metathesaurus is one example of a thesaurus. Statistics Canada maintains a thesaurus of preferred terms, synonyms, broader terms, and narrower terms for matters on which the government collects statistics, such as goods and services. This thesaurus is also bilingual English and French.
- An automatically derived thesaurus. Here, word co-occurrence statistics over a collection of documents in a domain are used to automatically induce a thesaurus; see Section 9.2.3.
- Query reformulations based on query log mining. Here, we exploit the manual query reformulations of other users to make suggestions to a new user. This requires a huge query volume, and is thus particularly appropriate to web search.

Thesaurus-based query expansion has the advantage of not requiring any user input. Use of query expansion generally increases recall and is widely used in many science and engineering fields. As well as such global analysis techniques, it is also possible to do query expansion by local analysis, for instance, by analyzing the documents in the result set. User input is now

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

► **Figure 9.8** An example of an automatically generated thesaurus. This example is based on the work in Schütze (1998), which employs latent semantic indexing (see Chapter 18).

usually required, but a distinction remains as to whether the user is giving feedback on documents or on query terms.

9.2.3 Automatic thesaurus generation

As an alternative to the cost of a manual thesaurus, we could attempt to generate a thesaurus automatically by analyzing a collection of documents. There are two main approaches. One is simply to exploit word cooccurrence. We say that words co-occurring in a document or paragraph are likely to be in some sense similar or related in meaning, and simply count text statistics to find the most similar words. The other approach is to use a shallow grammatical analysis of the text and to exploit grammatical relations or grammatical dependencies. For example, we say that entities that are grown, cooked, eaten, and digested, are more likely to be food items. Simply using word cooccurrence is more robust (it cannot be misled by parser errors), but using grammatical relations is more accurate.

The simplest way to compute a co-occurrence thesaurus is based on term-term similarities. We begin with a term-document matrix A, where each cell $A_{t,d}$ is a weighted count $w_{t,d}$ for term t and document d, with weighting so A has length-normalized rows. If we then calculate $C = AA^T$, then $C_{u,v}$ is a similarity score between terms u and v, with a larger number being better. Figure 9.8 shows an example of a thesaurus derived in basically this manner, but with an extra step of dimensionality reduction via Latent Semantic Indexing, which we discuss in Chapter 18. While some of the thesaurus terms are good or at least suggestive, others are marginal or bad. The quality of the associations is typically a problem. Term ambiguity easily introduces irrel-

evant statistically correlated terms. For example, a query for Apple computer may expand to Apple red fruit computer. In general these thesauri suffer from both false positives and false negatives. Moreover, since the terms in the automatic thesaurus are highly correlated in documents anyway (and often the collection used to derive the thesaurus is the same as the one being indexed), this form of query expansion may not retrieve many additional documents.

Query expansion is often effective in increasing recall. However, there is a high cost to manually producing a thesaurus and then updating it for scientific and terminological developments within a field. In general a domain-specific thesaurus is required: general thesauri and dictionaries give far too little coverage of the rich domain-particular vocabularies of most scientific fields. However, query expansion may also significantly decrease precision, particularly when the query contains ambiguous terms. For example, if the user searches for interest rate, expanding the query to interest rate fascinate evaluate is unlikely to be useful. Overall, query expansion is less successful than relevance feedback, though it may be as good as pseudo relevance feedback. It does, however, have the advantage of being much more understandable to the system user.

? **Exercise 9.7**
If A is simply a Boolean cooccurrence matrix, then what do you get as the entries in C?

9.3 References and further reading

Work in information retrieval quickly confronted the problem of variant expression which meant that the words in a query might not appear in a document, despite it being relevant to the query. An early experiment about 1960 cited by Swanson (1988) found that only 11 out of 23 documents properly indexed under the subject *toxicity* had any use of a word containing the stem *toxi*. There is also the issue of translation, of users knowing what terms a document will use. Blair and Maron (1985) conclude that "it is impossibly difficult for users to predict the exact words, word combinations, and phrases that are used by all (or most) relevant documents and only (or primarily) by those documents".

The main initial papers on relevance feedback using vector space models all appear in Salton (1971b), including the presentation of the Rocchio algorithm (Rocchio 1971) and the Ide dec-hi variant along with evaluation of several variants (Ide 1971). Another variant is to regard *all* documents in the collection apart from those judged relevant as nonrelevant, rather than only ones that are explicitly judged nonrelevant. However, Schütze et al. (1995) and Singhal et al. (1997) show that better results are obtained for routing by using only documents close to the query of interest rather than all

documents. Other later work includes Salton and Buckley (1990), Riezler et al. (2007) (a statistical NLP approach to RF) and the recent survey paper Ruthven and Lalmas (2003).

The effectiveness of interactive relevance feedback systems is discussed in (Salton 1989, Harman 1992, Buckley et al. 1994b). Koenemann and Belkin (1996) do user studies of the effectiveness of relevance feedback.

Traditionally Roget's thesaurus has been the best known English language thesaurus (Roget 1946). In recent computational work, people almost always use WordNet (Fellbaum 1998), not only because it is free, but also because of its rich link structure. It is available at: http://wordnet.princeton.edu.

Qiu and Frei (1993) and Schütze (1998) discuss automatic thesaurus generation. Xu and Croft (1996) explore using both local and global query expansion.