Relevance feedback and query expansion

Examining the failures of a straightforward retrieval system, it is usually obvious to a human that you will only be able to do high recall retrieval by also retrieving documents that refer to the same concept but which use different words. For example, you would want a search for aircraft to match plane (providing that it is an airplane, not a woodworking plane), and for a search on thermodynamics to match references to heat in appropriate discussions.

The methods for achieving this goal split into two major classes: global methods and local methods. Global methods are techniques for expanding or reformulating query terms which are independent of the query and results returned from it. The aim is 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)
- Query expansion via automatic thesaurus generation
- Techniques like spelling correction, which we discussed in Chapter 3

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

- Relevance feedback
- Pseudo-relevance feedback (also known as Blind relevance feedback)
- (Global) indirect relevance feedback

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

The idea of relevance feedback 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 not relevant.
- 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 the user 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

Figure 9.1 An example of relevance feedback searching over images. Here, the user enters an initial query. From http://nayana.ece.ucsb.edu/imsearch/imsearch.html (Newsam et al. 2001).
9.1 Relevance feedback and pseudo-relevance feedback

Figure 9.2 An example of relevance feedback searching over images (continued). Here the user views the initial query results.

Figure 9.3 An example of relevance feedback searching over images (continued). Here the user provides feedback on relevant results.
have difficulty formulating what they want in words, but can easily indicate relevant or non-relevant images. Figure 9.1 shows a user entering an initial query for bike on the demonstration system at:

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

The initial results returned are shown in Figure 9.2. In Figure 9.3, 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 9.4 then shows the new top-ranked results calculated after this round of relevance feedback.

Figure 9.5 shows a textual IR example where the user wishes to find out about new applications of space satellites.

9.1.1 The Rocchio Algorithm

The Rocchio Algorithm (Rocchio 1971) is the classic algorithm for implementing relevance feedback. It models a way of incorporating relevance feedback information into the vector space model of Chapter 7.

The underlying theory. We want to find a query vector that maximizes similarity with relevant documents while minimizing similarity with non-relevant documents. That is, if \( C_r \) is the set of relevant documents and \( C_{nr} \) is
9.1 Relevance feedback and pseudo-relevance feedback

Query: New space satellite applications

+ 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

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

* 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 9.5 Example of relevance feedback on a text collection. Following the initial query, the user marks some relevant documents (shown with a plus sign). The query is then expanded by 18 terms with weights as shown. The revised top results are then shown. A * marks the documents which were judged relevant in the relevance feedback phase.
Relevance feedback and query expansion

The Rocchio (1971) algorithm. This was the relevance feedback mechanism 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 irrelevant documents. The algorithm proposes using

\[ \tilde{q} = \arg \max_{\tilde{q}} [\text{sim}(q, C_r) - \text{sim}(q, C_{nr})] \]  

(9.1)

Under a model of cosine similarity, the optimal query vector \( \tilde{q}_{opt} \) for separating the relevant and non-relevant documents is:

\[ \tilde{q}_{opt} = \frac{1}{|C_r|} \sum_{d_j \in C_r} \tilde{d}_j - \frac{1}{|C_{nr}|} \sum_{d_j \in C_{nr}} \tilde{d}_j \]  

(9.2)

That is, the optimal query is the difference between the centroids of the relevant and non-relevant documents; see Figure 9.6. 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.

1. In the equation, \( \arg \max_x f(x) \) returns a value of \( x \) which maximizes the value of the function \( f(x) \).
9.1 Relevance feedback and pseudo-relevance feedback

The modified query $\vec{q}_m$:

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

where $\vec{q}_0$ is the original query vector, $D_r$ and $D_{nr}$ are the set of known relevant and non-relevant documents respectively, and $a$, $\beta$, and $\gamma$ 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 $\beta$ and $\gamma$. Starting from $\vec{q}_0$, the new query moves you some distance toward the centroid of the relevant documents and some distance away from the centroid of the non-relevant documents. This new query can be used for retrieval in the standard vector space model, as usual (see Chapter 7). Note that we can easily leave the positive quadrant of the vector space by subtracting off a non-relevant document’s vector. In the Rocchio algorithm, negative term weights are ignored. That is, the term weight is set to 0. Figure 9.7 shows the effect of applying relevance feedback.

Relevance feedback can improve both recall and precision. But, in practice, relevance feedback 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: in this situation, users can be expected to take time to review results and to iterate.

\[ \text{Figure 9.7} \] An application of Rocchio’s algorithm. Some documents have been labeled as relevant and non-relevant and the initial query vector is moved in response to this feedback.
on the search. It is also the case that positive feedback turns out to be much more valuable than negative feedback, and so people set $\gamma < \beta$. For example, reasonable values might be $\alpha = 1$, $\beta = 0.75$, and $\gamma = 0.15$. In fact, many systems, such as the image search system shown above, allow only positive feedback, which is equivalent to setting $\gamma = 0$. Another alternative, is to use only the one highest-ranked non-relevant document as negative feedback (so $|D_{nr}| = 1$ in Equation (9.3)). While many of the experimental results comparing various relevance feedback variants are rather inconclusive, some studies have suggested that this Ide dec-hi variant is the most effective or at least consistent (Ide 1971). In the other direction, another variant is to regard all documents in the collection apart from those judged relevant as non-relevant, rather than only ones that are explicitly judged non-relevant. 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.

**Exercise 9.1**

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

### 9.1.2 Probabilistic relevance feedback

Rather than reweighting the query in a vector space, if a user has told us some relevant and non-relevant documents, then we can proceed to build a classifier, such as with a Naive Bayes model (see Chapter 13). We can derive that the probability of a term $t_k$ appearing in a document depending on whether it is relevant or not (expressed by the variable $R$) is:

\[
P(t_k | R = \text{true}) = \frac{|D_{rk}|}{|D_r|}
\]

\[
P(t_k | R = \text{false}) = \frac{(N_k - |D_{rk}|)}{(N - |D_r|)}
\]

where $N$ is the total number of documents, $N_k$ is the number that contain $t_k$, and $D_{rk}$ is the number of known relevant documents containing $t_k$. Even though the set of known relevant documents is a perhaps small subset of the real set of relevant documents, assuming that the set of relevant documents is a small subset of the set of all documents, then the estimates given above will be reasonable.

At the end of the day, this gives another way of changing the query term weights. We will discuss such probabilistic approaches more in Chapters 11 and 13. But for the moment note that a disadvantage of this proposal is that the estimate uses only collection statistics and information about the term distribution within the documents judged relevant. It preserves 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. Rather one needs 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.

- 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 that relevance prototypes are well-behaved. 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 non-relevant 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, and relevant documents have significant vocabulary overlap, while similarities between relevant and irrelevant documents are small. Implicitly, the Rocchio relevance feedback model is treating relevant documents as a single cluster, which it models via the centroid of the cluster. So the approach does not work as well if there are several clusters. This can happen with:

- Subsets of the documents which use 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.
Relevance feedback and query expansion

9.4 Relevance Feedback on the Web

Some web search engines offer a similar/related pages feature. 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 search engine, which initially provided full relevance feedback. However, the feature was in time dropped, due to lack of use. This is probably in part the general observation that on the web almost nobody uses advanced search interfaces and would like to complete their search in a single interaction. But it 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 almost never concerned with getting sufficient recall.

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

Exercise 9.2

In Rocchio’s algorithm, what weight setting for $\alpha/\beta/\gamma$ does a “Find pages like this one” search correspond to?
9.1.5 Evaluation of relevance feedback strategies

Interactive relevance feedback can give very substantial gains in retrieval performance (Salton 1989, Harman 1992, Buckley et al. 1994).

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. And if you do this, you 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, measures of 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 judgements, 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. This scenario arises quite naturally in a document routing scenario where a user is assumed to have developed a query based on an existing document collection, and then the query will be applied to new documents as they come in. Such a scenario was explored in early TREC evaluations. Empirically, one round of relevance feedback is often very useful. Two rounds is sometimes marginally more useful.

9.1.6 Pseudo-relevance feedback

Pseudo-relevance feedback, also known as blind relevance feedback, provides a method for automatic local analysis. It attempts to automate the manual part of relevance feedback, so that one gets improved retrieval performance without requiring an extended interaction with the user. The method is to...
9 Relevance feedback and query expansion

<table>
<thead>
<tr>
<th>Term weighting</th>
<th>no RF</th>
<th>pseudo RF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inc.ltc</td>
<td>64.2%</td>
<td>72.7%</td>
</tr>
<tr>
<td>Lnu.ltu</td>
<td>74.2%</td>
<td>87.0%</td>
</tr>
</tbody>
</table>

Figure 9.8 Results showing pseudo relevance feedback greatly improving performance. These results are taken from the Cornell SMART system at TREC 4 (Buckley et al. 1996), and also contrast the use of two different length normalization schemes (L vs. l). Pseudo-relevance feedback consisted of adding 20 terms to each query.

Do normal retrieval to find an initial set of most relevant documents, to then assume that the top \( m \) 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. It has been found to improve performance in the TREC ad-hoc task. See for example the results in Figure 9.8. 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 that direction.

9.1.7 Indirect relevance feedback

One 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 judgements. 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 displayed summaries are equally good and about which parts of the page that the user has actually chosen to read. In the original DirectHit approach, this data about page goodness was gathered globally, rather than being user or query specific. This is one approach to the general area of clickstream mining.

9.1.8 Summary

Relevance feedback has been shown to be very effective at improving relevance of results. Requires enough judged documents, otherwise it’s unstable
(≥ 5 is recommended). Requires queries for which the set of relevant documents is medium to large. Full relevance feedback is painful for the user. Full relevance feedback is not very efficient in most IR systems. Other types of interactive retrieval may improve relevance by as much with less work.

Other uses of relevance feedback include:

• Following a changing information need
• Maintaining an information filter (e.g., for a news feed)
• Active learning (deciding which examples it is most useful to know the class of to reduce annotation costs).

9.2 Global methods for query reformulation

9.2.1 Vocabulary tools for query reformulation

Various user supports in the search process are very helpful for the user in knowing how their search is or isn’t working. This includes feedback 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. Another source of information is the suggestion of 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 (saying whether they are relevant/non-relevant), and this input is used to reweight the terms in the query for documents. In query expansion, users give additional input on query words or phrases, suggesting terms or saying whether they regard system suggestions as good or bad search terms. Figure 9.9 shows an example of query expansion options being presented in the Yahoo! web search engine. The central question in the use of query expansion is how to generate alternative or expanded queries for the user. The most common form of query expansion is global analysis, where by some means a form of thesaurus has been gathered. But it is also possible to do query expansion by local analysis, by analysing the documents in the result set. The distinction is again in whether the user is giving feedback on documents or the query terms.

Methods for building a thesaurus for query expansion include:
Figure 9.9 An example of query expansion in the interface of the Yahoo! web search engine in 2006. Note the expanded query suggestions appearing just below the “Search Results” bar.

- User query: cancer

- User query: skin itch

Figure 9.10 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.
Use of a controlled vocabulary that is maintained by editors. Here, there are canonical terms for concepts. The subject headings of traditional library subject indices, 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. Examples are shown in Figure 9.10.

A manual thesaurus. Here, people have built up sets of synonymous names for concepts. The UMLS metathesaurus is one example of a thesaurus. To take another, 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 lexicon.

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. For each term, \( t \), in a query, the query can be automatically expanded with synonyms and related words of \( t \) from the thesaurus. For example, in Figure 9.10, neoplasms was added to a search for cancer. Use of a thesaurus can be combined with ideas of term weighting: for instance, one might weight added terms less than original query terms. Use of query expansion generally increases recall and is widely used in many science/engineering fields. However, it 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. More seriously, there is a high cost to manually producing a thesaurus and then updating it for scientific changes. However, 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.

### 9.2.3 Automatic thesaurus generation

As an alternative to the cost of a manual thesaurus, one can attempt to generate a thesaurus automatically by analyzing a collection of documents. There
are two main approaches. One is simply to exploit word cooccurrence and to say that co-occurring words are likely to be similar. The other is to use a shallow grammatical analysis of the text and to exploit grammatical relations or grammatical dependencies. That is, we say that, for example, entities that are grown, cooked, eaten, and digested, are more likely to be food items. Simply using word cooccurrence is more robust, but using grammatical relations is more accurate.

The simplest way to compute a co-occurrence thesaurus is based on term-term similarities, which can be derived from a term-document matrix $A$ by calculating $C = AA^T$. If $A_{ij}$ has a perhaps normalized weighted count $w_{ij}$ for term $t_i$ and document $d_j$, then $C_{uv}$ has a similarity score between terms, with a larger number being better. Figure 9.11 shows an example of a thesaurus derived automatically in this way. While some of the thesaurus terms are good or at least quite suggestive, others are marginal or bad. The quality of the associations is quite typically a problem. Term ambiguity easily introduces irrelevant statistically correlated terms. A query for *Apple computer* may expand to *Apple red fruit computer*. In general one suffers from both false positives (words wrongly deemed similar) and false negatives. Moreover, since the terms in the automatic thesaurus are highly correlated in documents anyway (and often the same collection is used to derive the thesaurus as is being indexed), this form of query expansion may not retrieve many additional documents.

Query expansion is often effective in increasing recall. This is not always true with general thesauri, but it is fairly successful for subject-specific thesauri. In most cases, precision is decreased, often significantly. 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 9.3
If $A$ is simply a Boolean cooccurrence matrix, then what do you get as the entries in $C$?

9.3 References and further reading

The main initial papers on relevance feedback using vector space models all appear in Salton (1971). Later work includes Salton and Buckley (1990) and the recent survey paper Ruthven and Lalmas (2003).

Use of clickthrough data on the web to provide indirect relevance feedback is studied in more 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).