Future Directions

One area of research for query rewriting is how to detect query intent drifting after rewriting. For example, stemming query "marching network" to "march networks," or "blue steel" to "blued steel" is bad since the query after rewriting has different intent of the input. How to model intent drifting is a challenging task in Web search.

Cross-references

▶ Query Expansion
▶ Query Expansion models
▶ Query Rewriting
▶ Web Search Relevance Feedback

Recommended Reading

1. Anick P. Using terminological feedback for Web search refinement – a log-based study. In Proc. 26th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, 2003, pp. 88–95.
2. Brill E. and Moore R.C. An improved error model for noisy channel spelling correction. In Proc. 38th Annual Meeting of the Assoc. for Computational Linguistics, 2000, pp. 86–293.
3. Croft W.B. and Harper D.J. Using probabilistic models of document retrieval without relevance information. J. Doc., 35 (4):285–295, 1979.
4. Cucerzan S. and Brill E. Spelling correction as an iterative process that exploits the collective knowledge of Web users. In Proc. Conf. on Empirical Methods in Natural Language Processing, 2004, pp. 293–300.
5. Cui H., Wen J.R., Nie J.Y., and Ma W.Y. Probabilistic query expansion using query logs. In Proc. 11th Int. World Wide Web Conference, 2002, pp. 325–332.
6. Fonseca B.M., Golgher P., Pssas B., Ribeiro-Neto B., and Ziviani N. Concept-based interactive query expansion. In Proc. 14th ACM Int. Conf. on Information and Knowledge Management, 2008, pp. 696–703.
7. Jansen B.J., Spink A., and Saracevic T. Real life, real users, and real needs: a study and analysis of user queries on the web. Inf. Process. Manage. Int. J., 36(2):207–227, 2000.
8. Joachims T., Granka L., Pang B., Hembrooke H., and Gay G. Accurately interpreting clickthrough data as implicit feedback. In Proc. 31st Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, 2005, pp. 154–161.
9. Jones R. and Fain D. Query word deletion prediction. In Proc. 26th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, 2003, pp. 435–436.
10. Jones R., Rey B., Madani O., and Greiner W. Generating query substitutions. In Proc. 15th Int. World Wide Web Conference, 2006, pp. 387–396.
11. Lovins J.B. Development of a stemming algorithm. Mech. Translat. Comput. Ling., 2:22–31, 1968.
12. Peng F., Ahmed N., Li X., and Lu Y. Context sensitive stemming for Web search. In Proc. 33rd Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, 2007, pp. 639–646.
13. Porter M.F. An algorithm for suffix stripping. Program, 14(3):130–137, 1980.
14. Qiu Y. and Frei H.P. Concept based query expansion. In Proc. 16th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, 1993, pp. 160–169.
15. Tan B. and Peng F. Unsupervised query segmentation using generative language models and wikipedia. In Proc. 17th Int. World Wide Web Conference, 2008, pp. 347–356.
16. Wei X., Peng F., and Dumoulin B. Analyzing Web text association to disambiguate abbreviation in queries. In Proc. 34th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, 2008, pp. 751–752.
17. Xu J. and Croft B. Query expansion using local and global document analysis. In Proc. 19th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, 1996, pp. 4–11.

Web Search Relevance Feedback

Hui Fang[1], ChengXiang Zhai[2]
[1]University of Delaware, Newark, DE, USA
[2]University of Illinois at Urbana-Champaign, Urbana, IL, USA

Definition

Relevance feedback refers to an interactive cycle that helps to improve the retrieval performance based on the relevance judgments provided by a user. Specifically, when a user issues a query to describe an information need, an information retrieval system would first return a set of initial results and then ask the user to judge whether some information items (typically documents or passages) are relevant or not. After that, the system would reformulate the query based on the collected feedback information, and return a set of retrieval results, which presumably would be better than the initial retrieval results. This procedure could be repeated.

Historical Background

Quality of retrieval results highly depends on how effective a user's query (usually a set of keywords) is in distinguishing relevant documents from non-relevant ones. Ideally, the keywords used in the query should occur only in the relevant documents and not in any non-relevant document. Unfortunately, in reality, it is often difficult for a user to come up with good keywords, mainly because the same concept can be described using different words and the user often has no clue about which words are actually used in the relevant documents of a collection. A main motivation for relevance feedback comes from the observation that although it may be difficult for a user to formulate a good query, it is often much easier for the user to judge whether a document or a passage is relevant.

Relevance feedback was first studied in the vector space model by Rocchio [8]. After that, many other relevance feedback methods have been proposed and studied in other retrieval models [6,15], including classical probabilistic models and language modeling approach. Although these methods are different, they all try to make the best use of the relevance judgments provided by the user and generally rely on some kind of learning mechanism to learn a more accurate representation of the user's information need. Relevance feedback has proven to be one of the most effective methods for improving retrieval performance [8,6,10].

Unfortunately, due to the extra effort that a user must make in relevance feedback, users are often reluctant to provide such explicit relevance feedback. When there are no explicit relevance judgments available, *pseudo feedback* may be performed, also known as *blind relevance feedback* [2,7]. In this method, a small number of top-ranked documents in the initial retrieval results are assumed to be relevant, and relevance feedback is then applied. This method also tends to improve performance on average (especially recall). However, pseudo feedback method does not always work well, especially for queries where none of the top N documents is relevant. The poor performance is expected because the non-relevant documents are assumed to be relevant, which causes the reformulated query to shift away from the original information need.

Somewhere inbetween relevance feedback and pseudo feedback is a technique called *implicit feedback* [5,4,11]. In implicit feedback, a user's actions in interacting with a system (e.g., clickthroughs) are used to infer the user's information need. For example, a result viewed by a user may be regarded as relevant (or more likely relevant than a result skipped by a user). Search logs serve as the primary data for implicit feedback; their availability has recently stimulated a lot of research in learning from search logs to improve retrieval accuracy (e.g., [3]).

Relevance feedback bears some similarity to query expansion in the sense that both methods attempt to select useful terms to expand the original query for improving retrieval performance. However, these two methods are not exactly same. First, query expansion does not necessarily assume the availability of relevance judgments. It focuses on identifying useful terms that could be used to further elaborate the user's information need. The terms can come from many different sources, not just feedback documents (often called local methods for query expansion). For example, they can also come from any document in the entire collection (called global methods) [14] or external resources such as a thesaurus. Second, while relevance feedback is often realized through query expansion, it can be otherwise. For example, when all available examples are non-relevant, other techniques than query expansion may be more appropriate [13].

Foundations

The basic procedure for relevance feedback includes the following steps. A user first issues a query, and the system returns a set of initial retrieval results. After returning the initial results, the user is asked to judge whether the presented information items (e.g., documents or passages) are relevant or non-relevant. Finally, the system revises the original query based on the collected relevance judgments typically by adding additional terms extracted from relevant documents, promoting weights of terms that occur often in relevant documents, but not so often in non-relevant documents, and down-weighting frequent terms in non-relevant documents. The revised query is then executed and a new set of results is returned. The procedure can be repeated more than once.

Technically, relevance feedback is a learning problem in which one learns from the relevant and non-relevant document examples how to distinguish new relevant documents from non-relevant documents. The learning problem can be cast as to learn either a binary classifier that can classify a document into relevant vs. non-relevant categories or a ranker that can rank relevant documents above non-relevant documents. Thus in principle, any standard supervised learning methods can be potentially applied to perform relevance feedback. However, as a learning problem, relevance feedback poses several special challenges, and as a result, a direct application of a standard supervised learning method is often not very effective. First, many supervised learning methods only work well when there are a relatively large number of training examples, but in the case of relevance feedback, the number of positive training examples is generally very small. Second, the training examples in relevance feedback are usually from the top-ranked documents, thus they are not a random sample and can be biased. Third, it is unclear how to handle the original query. The query may be treated as a special short example of relevant documents, but intuitively this special relevant example is much more important than other relevant examples and the terms in the query should be sufficiently emphasized to avoid drifting away from the original query concept. These challenges are exacerbated in the case of pseudo feedback as the query would be the only reliable relevance information provided by the user.

In the information retrieval community, many relevance feedback algorithms have been developed for different retrieval models. A commonly used feedback algorithm in vector space models is the *Rocchio algorithm* developed in early 1970s [8], which remains a robust effective state-of-the-art method for relevance feedback. In Rocchio, the problem of relevance feedback is defined as finding an optimal query to maximize its similarity to relevant documents and minimize its similarity to non-relevant documents. The revised query can be computed as

$$Q_r = \alpha \cdot Q_o + \frac{\beta}{|D_r|} \cdot \sum_{d_i \in D_r} d_i - \frac{\gamma}{|D_n|} \cdot \sum_{d_j \in D_n} d_j,$$

where Q_r is the revised query vector, Q_o is the original query vector, D_r and D_n are the sets of vectors for the known relevant and non-relevant documents, respectively. α, β, and γ are the parameters that control the contributions from the two sets of documents and the original query. Empirical results show that positive feedback is much more effective than negative feedback, so the optimal value of γ is often much smaller than that of β. A term would have a higher weight in the new query vector if it occurs frequently in relevant documents but infrequently in non-relevant documents. The IDF weighting further rewards a term that is rare in the whole collection.

The feedback method in classical probabilistic models is to select expanded terms primarily based on *Robertson/Sparck-Jones weight* [6] defined as follows:

$$w_i = log \frac{r_i/(R - r_i)}{(n_i - r_i)/(N - n_i - R + r_i)},$$

where w_i is the weight of term i, r_i is the number of relevant documents containing term i, n_i is the number of documents containing term i, R is the number of relevant documents in the collection, and N is the number of documents in the collection.

In language modeling approaches, feedback can be achieved through updating the query language model based on feedback information, leading to *model-based feedback methods*[15]. Relevance feedback received little attention in logical model, but some possible feedback methods have been discussed in [9].

Despite the difference in their way of performing relevance feedback, all these feedback methods implement the same intuition, which is to improve query representation by introducing and rewarding terms that occur frequently in relevant documents but infrequently in non-relevant documents. When optimized, most of these methods tend to perform similarly. Improvements over these basic feedback methods include (1) *passage feedback* [1] which can filter out the non-relevant part of a long relevant document in feedback, (2) *query zone* [12] which can improve the use of negative feedback information, and (3) *pure negative feedback* [13] which aims at exploiting purely negative information to improve performance for difficult topics.

Most studies focus on how to use the relevance judgments to improve the retrieval performance. The issue of choosing optimally documents to judge for relevance feedback has received considerably less attention. Recent studies on active feedback [11] have shown that choosing documents with more diversity for feedback is a better strategy than choosing the most relevant documents for feedback in the sense that the relevance feedback information collected with the former strategy is more useful for learning. However, this benefit may be at the price of sacrificing the utility of presented documents from a user's perspective.

Although relevance feedback improves retrieval accuracy, the improvement comes at the price of possibly slowing down the retrieval speed. Indeed, virtually all relevance feedback algorithms involve iterating over all the terms in all the judged examples, so the computational overhead is not negligible, making it a challenge to use relevance feedback in applications where response speed is critical.

Key Applications

Relevance feedback is a general effective technique for improving retrieval accuracy for all search engines, which is applicable when users are willing to provide explicit relevance judgments. Relevance feedback is particularly useful when it is difficult to completely and precisely specify an information need with just keywords (e.g., *multimedia search*); in such a case, relevance feedback can be exploited to learn features that can characterize the information need from the judged examples (e.g., useful non-textual features in multimedia search) and combine such features with the original query to improve retrieval results. Relevance feedback is also a key technique in *personalized search* where both explicit and implicit feedback information could be learned from all the collected information about a user to improve the current search results for the user.

Despite the fact that relevance feedback is a mature technology, it is not a popular feature provided by the current web search engines possibly because of the computational overhead. However, the feature "finding similar pages" provided by Google can be regarded as relevance feedback with just one relevant document.

Future Directions

Although relevance feedback has been studied for decades and many effective methods have been proposed, it is still unclear what is the optimal way of performing relevance feedback. The recent trend of applying machine learning to information retrieval, especially research on learning to rank and statistical language models, will likely shed light on the answer to this long-standing question.

So far, research on relevance feedback has focused on cases where at least several relevant examples can be obtained in the top-ranked documents. However, when a topic is difficult, a user may not see any relevant document in the top-ranked documents. In order to help such a user, negative feedback (i.e., relevance feedback with only non-relevant examples) must be performed. Traditionally negative feedback has not received much attention due to the fact that when relevant examples are available, negative feedback information tends not to be very useful. An important future research direction is to study how to exploit negative feedback to improve retrieval accuracy for difficult topics as argued in [13].

Active feedback [11] is another important topic worth further studying. Indeed, to minimize a user's effort in providing explicit feedback, it is important to select the most informative examples for a user to judge so that the system can learn most from the judged examples. It would be interesting to explore how to apply/adapt many active learning techniques developed in the machine learning community to perform active feedback.

Experimental Results

The effectiveness of a relevance feedback method is usually evaluated using the standard information retrieval evaluation methodology and test collections. A standard test collection includes a document collection, a set of queries and judgments indicating whether a document is relevant to a query. The initial retrieval results are compared with the feedback results to show whether feedback improves performance. The performance is often measured using Mean Average Precision (MAP) which reflects the overall ranking accuracy or precision at top k (e.g., top 10) documents which reflects how many relevant documents a user can expect to see in the top-ranked documents.

When comparing different relevance feedback methods, the amount of feedback information is often controlled so that every method uses the same judged documents. The judged or seen documents are usually excluded when computing the performance of a method to more accurately evaluate the performance of a method on *unseen* documents. The evaluation is significantly more challenging when the amount of feedback information can not be controlled, such as when different strategies for selecting documents for feedback are compared or a feedback method and a non-feedback method are compared. In such cases, it is tricky how to handle the judged documents. If they are not excluded when computing the performance of a method, the comparison would be unfair and favor a feedback that has received more judged examples. However, if they are excluded, the performance would not be comparable either because the test set used to compute the performance of each method would likely be different and may contain a different set of relevant and non-relevant documents.

Data Sets

Many standard information retrieval test collections can be found at: http://trec.nist.gov/

URL to Code

Lemur project contains the code for most existing relevance feedback method, which can be found at: http://lemurpoject.org/.

Cross-references

▶ Implicit Feedback
▶ Pseudo Feedback
▶ Query Expansion

Recommended Reading

1. Allan J. Relevance feedback with too much data. In Proc. 18th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, 1995, pp. 337–343.
2. Buckley C. Automatic query expansion using SMART: TREC-3. In Overview of the Third Text Retrieval Conference (TREC-3), D. Harman (ed.), 1995, pp. 69–80.
3. Burges C., Shaked T., Renshaw E., Lazier A., Deeds M., Hamilton N., and Hullender G. Learning to rank using gradient descent. In Proc. 22nd Int. Conf. on Machine Learning, 2005, pp. 89–96.
4. Joachims T. Optimizing search engines using clickthrough data. In Proc. 8th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, 2002, pp. 133–142.
5. Kelly D. and Teevan J. Implicit feedback for inferring user preference. SIGIR Forum, 37(2):18–28, 2003.
6. Robertson S.E. and Jones K.S. Relevance weighting of search terms. J. Am. Soc. Inf. Sci., 27(3):129–146, 1976.
7. Robertson S.E., Walker S., Jones S., Hancock-Beaulieu M.M., and Gatford M. Okapi at TREC-3. In Proc. The 3rd Text Retrieval Conference, 1995, pp. 109–126.
8. Rocchio J. Relevance feedback in information retrieval. In The SMART Retrieval System: Experiments in Automatic Document Processing. Prentice-Hall, Englewood Cliffs, NJ, 1971, pp. 313–323.
9. Ruthven I. and Lalmas M. A survey on the use of relevance feedback for information access system. Knowl. Eng. Rev., 18(2):95–145, 2003.
10. Salton G. and Buckley C. Improving retrieval performance by relevance feedback. J. Am. Soc. Inf. Sci., 44(4):288–297, 1990.
11. Shen X. and Zhai C. Active feedback in ad hoc information retrieval. In Proc. 31st Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, 2005, pp. 59–66.
12. Singhal A., Mitra M., and Buckley C. Learning routing queries in a query zone. In Proc. 20th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, 1997, pp. 25–32.
13. Wang X., Fang H., and Zhai C. A study of methods for negative relevance feedback. In Proc. 34th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, 2008, pp. 219–226.
14. Xu J. and Croft W.B. Query expansion using local and global document analysis. In Proc. 19th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, 1996, pp. 4–11.
15. Zhai C. and Lafferty J. Model-based feedback in the language modeling approach to information retrieval. In Proc. Int. Conf. on Information and Knowledge Management, 2001, pp. 403–410.

Web Search Relevance Ranking

Hugo Zaragoza[1], Marc Najork[2]
[1]Yahoo! Research, Barcelona, Spain
[2]Microsoft Research, Mountain View, CA, USA

Synonyms

Ranking; Search ranking; Result ranking

Definition

Web search engines return lists of web pages sorted by the page's relevance to the user query. The problem with web search relevance ranking is to estimate relevance of a page to a query. Nowadays, commercial web-page search engines combine hundreds of features to estimate relevance. The specific features and their mode of combination are kept secret to fight spammers and competitors. Nevertheless, the main types of features at use, as well as the methods for their combination, are publicly known and are the subject of scientific investigation.

Historical Background

Information Retrieval (IR) Systems are the predecessors of Web and search engines. These systems were designed to retrieve documents in curated digital collections such as library abstracts, corporate documents, news, etc. Traditionally, IR relevance ranking algorithms were designed to obtain high recall on medium-sized document collections using long detailed queries. Furthermore, textual documents in these collections had little or no structure or hyperlinks. Web search engines incorporated many of the principles and algorithms of Information Retrieval Systems, but had to adapt and extend them to fit their needs.

Early Web Search engines such as Lycos and AltaVista concentrated on the scalability issues of running web search engines using traditional relevance ranking algorithms. Newer search engines, such as Google, exploited web-specific relevance features such as hyperlinks to obtain significant gains in quality. These measures were partly motivated by research in citation analysis carried out in the bibliometrics field.