Relevance Feedback for Best Match Term Weighting Algorithms in Information Retrieval

Djoerd Hiemstra
Microsoft Research, Cambridge, UK
and University of Twente, Enschede, The Netherlands
hiemstra@cs.utwente.nl

Stephen Robertson
Microsoft Research, Cambridge, UK
and City University London, U.K.
ser@microsoft.com

Abstract Personalisation in full text retrieval or full text filtering implies reweighting of the query terms based on some explicit or implicit feedback from the user. Relevance feedback inputs the user's judgements on previously retrieved documents to construct a personalised query or user profile. This paper studies relevance feedback within two probabilistic models of information retrieval: the first based on statistical language models and the second based on the binary independence probabilistic model. The paper shows the resemblance of the approaches to relevance feedback of these models, introduces new approaches to relevance feedback for both models, and evaluates the new relevance feedback algorithms on the TREC collection. The paper shows that there are no significant differences between simple and sophisticated approaches to relevance feedback.

1 Introduction

Relevance feedback in full text information retrieval inputs the user's judgements on previously retrieved documents to construct a personalised query. These algorithms utilise the distribution of terms over relevant and irrelevant documents to re-estimate the query term weights, resulting in an improved user query. Relevance feedback is especially helpful in applications where users have a long-lasting information need, with plenty of opportunity to give feedback to the system, for instance in adaptive filtering systems [6].

The theory of relevance feedback algorithms is well-developed for the traditional vector space model [15], and for the traditional binary independence probabilistic model [12]. However, the binary independence probabilistic model only produces a partial ranking of the retrieved set of documents. For extensions of the probabilistic model that do produce a full ranking of the documents [13] [16], the old theory is no longer appropriate. Algorithms that produce a full ranking of the documents are called *best match* retrieval algorithms. Probabilistic best match retrieval algorithms were recently proposed by using statistical language models [4] [8] [9] [10], but these models also lack a well-founded approach to relevance feedback.

This paper introduces new relevance feedback algorithms for both probabilistic approaches to information retrieval mentioned above: the language models and the binary independence model. It introduces a new relevance feedback algorithm for language model-based information retrieval systems by utilising the expectation maximisation (EM-)algorithm [1]. The new relevance feedback algorithm for the binary independence model is a generalisation of the traditional Robertson/Sparck-Jones relevance weight [12]. It calculates the expected weight over the ranked list of documents retrieved by a single term query. We argue that the traditional Roberston/Sparck-Jones relevance weight [12] is not appropriate for best match versions of the model, like for instance the BM25 algorithm [13].

This paper is organised as follows. Section 2 introduces the language model-based retrieval algorithms, including the EM-algorithm for relevance feedback. Section 3 describes the binary independence model and introduces the new relevance feedback algorithm for best match versions of the model. Section 4 gives experimental results of both relevance feedback models, using a test collection from the Text Retrieval Conferences (TREC). Finally, Section 5 contains concluding remarks.

2 Using language models for relevance feedback

When using language models for retrieval, one builds a simple language model for each document in the collection. Given a query, documents are ranked by the probability that the language model of each document generated the query. The paper follows the language models introduced in [3] by modelling the probability that a query $T_1, T_2,..., T_n$ of length n is sampled from a document with document identifier D as follows.

(1) $$P(D) \cdot P(T_1, T_2, \cdots, T_n \mid D) = P(D) \cdot \prod_{i=1}^{n} ((1 - l_i) P(T_i) + l_i P(T_i \mid D))$$

Similar models are introduced in [8], [9] and [10]. With the exception of [10] they all use the linear interpolation of global and local probabilities. Probabilities might be estimated as in equation (2), where the term frequency $tf(t_i, d)$ is the number of time the term t_i occurs in a document d, and the document frequency $df(t_i)$ is the number of documents in which the term t_i occurs.

(2) $$P(T_i = t_i) = \frac{df(t_i)}{\sum_t df(t)} \quad , \quad P(T_i = t_i \mid D = d) = \frac{tf(t_i, d)}{\sum_t tf(t, d)} \quad , \quad P(D = d) = \frac{\sum_t tf(t, d)}{\sum_t \sum_d tf(t, d)}$$

In equation (1), l_i is an unknown parameter, denoting the probability that the term on position i in the query is important. It is easy to verify for each term T_i that if l_i = 0 the term does not have any influence on the ranking of the document, whereas for l_i = 1 the term is mandatory in the retrieved documents, i.e. documents that do not contain the term are assigned zero probability. So, for l_i = 0 the term is definitely unimportant, whereas for l_i = 1 the term is definitely important. A naive approach to relevance feedback would estimate the value of l_i directly from the occurrences of terms in relevant documents as follows.

(3) $$l_i = \frac{r_i}{R}$$

In equation (3), r_i is the number of documents in which t_i occurs, and R is the total number of known relevant documents. There is however no guarantee that equation (3) optimises system performance, because l_i refers to an unknown, hidden event (see e.g. [8]), which cannot be observed from the occurrences of terms in the relevant documents alone. A standard procedure for estimating probabilities of unknown parameters from incomplete data is the expectation maximisation algorithm (EM-algorithm). The algorithm iteratively maximises the probability of the query t_1, t_2,..., t_n given R relevant documents d_1, d_2,..., d_R. The resulting EM-algorithm is defined as follows [7] [9].

(4) $$\begin{aligned} &\text{E-step:} \quad r_i' = \sum_{j=1}^{R} \frac{l_i^{(p)} P(T_i = t_i \mid D_j = d_j)}{(1 - l_i^{(p)}) P(T_i = t_i) + l_i^{(p)} P(T_i = t_i \mid D_j = d_j)} \\ &\text{M-step:} \quad l_i^{(p+1)} = \frac{r_i'}{R} \end{aligned}$$

The expectation step calculates the expected number of documents in which t_i is important. The maximisation step simply involves a maximum likelihood estimate similar to equation (3). The EM-algorithm should be used by initialising the relevance weights to some initial value, e.g. $l_i^{(0)}$ = 0.5 and then iterate through the E-step and M-step until the value of l_i does not change significantly anymore (p denotes the iteration number).

3 The binary independence probabilistic model

The binary independence assumption states that, given relevance L (and irrelevance $\bar{L}$), the attributes A_i of a document D are statistically independent. In the case of text retrieval, the document attributes are simply the terms in the documents. In the traditional model, the value of a document attribute is either 1, meaning the term is present in the document, or 0, the term is absent. Symbolically, the binary independence assumption is [12]:

(5) $$O(L \mid D) = O(L) \cdot \prod_{i=1}^{l} \frac{P(A_i \mid L)}{P(A_i \mid \bar{L})}$$

In the formula l is the number of non-equal terms in the query, and $O(L \mid D)$ is the probability odds of relevance of a document: $O(L \mid D) = P(L \mid D) / (1 - P(D))$. Ranking the document by $O(L \mid D)$ will in fact rank the documents by their probability of relevance. The probabilities $P(A_i \mid L)$ and $P(A_i \mid \bar{L})$ can be estimated if some relevant (and irrelevant) documents are known, i.e. if some of the preferences of the user are known to the system. If R is the number of relevant documents, N is the number of retrieved documents, r_i is the number of relevant documents in which the term is present, and n_i is the number of retrieved documents in which the term is present, then the probabilities are defined as:

(6) $$P(A_i \mid L) = \frac{r_i}{R} \quad \text{and} \quad P(A_i \mid \bar{L}) = \frac{n_i - r_i}{N - R}$$

The traditional probabilistic model is motivated by Robertson’s probability ranking principle [11], which states that, if documents are ranked by decreasing probability of relevance, then the overall effectiveness of the system to its users will be the best that is obtainable on the basis of the data available to the system. This can be shown by giving the measures of retrieval effectiveness a probabilistic interpretation. In fact, the performance measures *recall* and *fallout* are respectively defined by the first and second formula in equation (6), if the query was the single term corresponding with A_i. Furthermore, Robertson showed that *recall*, *fallout* and *precision* are related just as the corresponding probabilities are related to the probability of relevance. If the binary independence assumption holds, then the traditional probabilistic model will in fact optimise system performance, because:

$$\text{(7)} \qquad \text{expected precision} = O(L) \cdot \frac{\text{expected recall}}{\text{expected fallout}}$$

The probabilities calculated with equation (5) are used to define a ranking of the documents in the collection. Any order preserving transformation of the probabilities will be as useful as the probabilities themselves. The following formula defines the same ranking as (5), but uses only the matching terms in its computation, as it assigns a zero weight to the non-matching terms.[1]

$$\text{(8)} \qquad O(L\,|\,D) \propto \sum_{i \in \text{matching terms}} w_i, \qquad w_i = \log \frac{P(A_i\,|\,L)(1-P(A_i\,|\,\overline{L}))}{P(A_i\,|\,\overline{L})(1-P(A_i\,|\,L))}$$

Equation (8) is called the Robertson/Sparck-Jones weight [12]. Recently, Robertson and Walker [13], proposed an extension of the traditional probabilistic model, in which the document attributes are no longer binary valued. Instead, the attributes might be any natural number, indicating the number of times the attribute occurs in the document. One of the resulting best match algorithms, BM25, is defined as follows [16]:

$$\text{(9)} \qquad BM25\!: \; O(L\,|\,D) \propto \sum_{i \in \text{matching terms}} QTF_i \cdot \frac{(k_1+1)\cdot TF_i}{k_1((1-b)+b\,\frac{DL}{AVDL})+TF_i} \cdot w_i$$

The parameters k_1 and b are tuning constants, which values should be set experimentally. We adopted the notation from [15]. The following notations refer to the same quantities: TF_i and $tf(t_i, d)$ refer to the number of times a term occurs in a document; DL and $\Sigma_t\, tf(t, d)$ refer to the length of a document; QTF_i refers to the number of times a term occurs in the query; $AVDL$ is the average document length in the collection.

The BM25 algorithm is motivated by some clever approximations to the 2-Poisson model [2], but makes a rather large step in using the Robertson/Sparck-Jones weight w_i (see equation (8)) to replace an equivalent function relating eliteness to relevance. Like the probability of term importance of the language modelling approach of section 2, the probability of term eliteness of the 2-Poisson model is an unknown parameter which event cannot simply be observed by mere term occurrence. Note that the Robertson/Sparck-Jones weight is based on the fact that the result a single term is an unranked list of documents. However, if the query was the single term corresponding to A_i, then the result set of the BM25 algorithm will not consist of an unranked list of documents. Posing a single-term query to a BM25 system will produce a *ranked* list of documents. At different cut-off levels we can expect different recall and fallout values. Therefore fixed proportions can no longer calculate the expected recall and the expected fallout. We therefore reject the hypothesis that relevance feedback procedures based on the term occurrences in relevant and non-relevant documents (i.e. equation (6)) optimises retrieval performance of the BM25 algorithm in terms of the probability ranking principle.

For ranked sets of retrieved documents, it is standard practice to average the performance measures like precision, recall and fallout over different cut-off levels. A simple standard (within the TREC conferences) approach to the calculation of expected precision is the average precision measure:

$$\text{average precision} = \Big(\sum_{k \in \text{ranks of relevant docs.}} \text{precision at } k\Big) \,/\, R$$

This is indeed expected precision if we assume that the probability $P(k)$ of a user looking for k relevant documents is uniformly distributed for $1 \leq k \leq R$. Under the same assumption we might define the expected recall/fallout ratio to be:

$$\text{expected recall/fallout ratio} = \Big(\sum_{k \in \text{ranks of relevant docs.}} \frac{\text{recall at } k}{\text{fallout at } k}\Big) \,/\, R$$

According to equation (7), precision on each cut-off level will be maximised if the recall/fallout ratio is maximised. Maximising average precision requires therefore, the maximisation of the average (or expected) recall/fallout ratio. The resulting measure optimises retrieval performance of bet match retrieval versions of the binary independence model. Following the order preserving transformation of equation (8), the expected Robertson/Sparck-Jones (RS) weight only has to be applied to the matching terms:

[1] a similar order preserving transformation exist for the language models ranking algorithm [5].

$$(10) \qquad \text{expected RS weight} = \log\Big(\Big(\sum_{k \in \text{ranks of relevant docs.}} \frac{(\text{recall at } k)(1-\text{fallout at } k)}{(\text{fallout at } k)(1-\text{recall at } k)}\Big) / R\Big)$$

Under the assumption that a user is equally likely to look for any k, $1 \leq k \leq R$ documents, equation (10) will (on average) lead to optimum retrieval performance on any document cut-off level. If for a single query term i the retrieved set is unranked, the user has to look at all of the documents at once or at none at all, and the Robertson/Sparck-Jones weight will be equal to its original definition.[2]

When calculating the average precision, there is usually a cut-off level below which the precision is assumed to be zero. For practical reasons the TREC evaluations lowest cut-off level to calculate average precision is set to be a thousand documents. For practical reasons, we might also define a cut-off level below which the recall/fallout and the Robertson/Sparck-Jones weight are assumed to be constant. In this case, we would assume that both recall and fallout approach roughly equal values and therefore that the recall/fallout ratio and the Robertson/Sparck-Jones weight are both 1 for each relevant document below the lowest cut-off level. A practical lowest cut-off level would be n_i: the number of documents containing the term i.

4 Experimental results

This section sums up experimental results of the relevance feedback algorithms discussed in this paper. Experiments were done with the Okapi Basic Search System (BSS) [14]. All experiments used topics 401-450 of the main TREC collection [17]. Each experiment was repeated using different fields of the TREC topics (title, description and narrative), in order to construct short, medium and long queries.

Ad-hoc retrieval

Ad-hoc retrieval mimics the situation in which a user enters an initial query, that is, when there are no previously retrieved documents to guide the search. The ad-hoc experiments serve as the base line for retrieval performance. After relevance feedback, we expect retrieval performance to go up. The following tables show different performance measures of one experiment. We will compare different methods by their average precision displayed in the last column. Precision at 10, 30 and 100 documents is given as additional information and will show if a method for instance affects high precision while not affecting average precision.

experiment	prec. at 10	prec. at 30	prec. at 100	average prec.
LM short queries	0.4660	0.3647	0.2338	0.2595
LM medium queries	0.4860	0.3893	0.2396	0.2818
LM long queries	0.5180	0.3867	0.2430	0.2879
BM25 short queries	0.4660	0.3433	0.2282	0.2390
BM25 medium queries	0.4700	0.3607	0.2346	0.2608
BM25 long queries	0.4760	0.3720	0.2400	0.2618

Table 1 ad-hoc retrieval runs

Table 1 shows the results of the language models (LM) algorithm use $l = 0.2$, and the BM25 algorithm, using $k_1 = 1.2$ and $b = 0.75$. Pair wise comparison of the average precision of the LM algorithm and the BM25 shows that the differences are not statistically significant at the 1 % level according to a two-sided sign test.

Relevance feedback

The Relevance feedback experiments reported in this paper are retrospective relevance weighting experiments similar to the ones described by Robertson and Sparck-Jones [12] and recently by Sparck-Jones et al. [15]. In these experiments, the system is presented the full list of all relevant documents. This is in fact an unrealistic scenario: In practice a predictive relevance weighting task, where the system is presented only a few of the full list of relevance documents, will be much more interesting. From a theoretical point of view, however, the retrospective task *is* interesting. If the algorithms investigated obey the probability ranking principle, then the best overall effectiveness of the system will be achieved on basis of the data that is available to the system. If *all* relevance information is made available to the system, then we expect the system to achieve optimal performance. Of course, we expect the system never to decrease the performance of a query. If the system

[2] this is different from standard practice in evaluation, where documents with equal scores are often sorted by document-id

decreases the performance of a query in the retrospective relevance weighting task, then it ‘behaves irrationally’: it is a clear indication that it violates the probability ranking principle.

experiment	prec. at 10	prec. at 30	prec. at 100	average precision
LM short queries	0.4920	0.3800	0.2512	0.2767 (+ 6.6 %)
LM medium queries	0.5200	0.4087	0.2614	0.3147 (+ 11.7 %)
LM long queries	0.5960	0.4427	0.2756	0.3432 (+ 19.2 %)
BM25 short queries	0.4780	0.3560	0.2394	0.2545 (+ 6.5 %)
BM25 medium queries	0.5420	0.4033	0.2610	0.2981 (+ 14.3 %)
BM25 long queries	0.5980	0.4400	0.2810	0.3245 (+ 23.9 %)

Table 2 simple relevance feedback algorithms

The simple relevance feedback experiments (defined by equation (3) and equation (6)) are presented in Table 2. The last column also displays the relative increase in average precision compared to the ad-hoc runs of Table 1. The table shows that both the LM and the BM25 algorithms show similar increase in performance. The relative performance increase on long queries is somewhat better for the BM25 algorithm. Pair wise comparison of the ad-hoc experiments and the simple relevance experiments shows that both algorithms show a decrease in performance for some queries. For the LM algorithm, the differences are significant at the 1 % level according to a two-sided sign test. For the BM25 algorithm, the differences between ad-hoc and relevance feedback of the short and medium queries are not statistically significant. Clearly, the BM25 algorithm behaves irrationally in this respect.

The working hypothesis of this paper is that the simple relevance feedback algorithms can be improved upon by taking the number of occurrences of terms in documents into account. The results of the well-motivated relevance feedback algorithms are presented in Table 3. The LM-algorithm shows small but consistent improvement over the previous experiments. The BM25 algorithm only shows improvement for the short and medium-sized queries. Especially the high precision of the short queries seems to be noticeably better than before. For long queries, however, the results are slightly worse than the ordinary Robertson/Sparck-Jones algorithm.

experiment	prec. at 10	prec. at 30	prec. at 100	average precision
LM short queries	0.4980	0.3813	0.2514	0.2781 (+ 7.2 %)
LM medium queries	0.5280	0.4160	0.2634	0.3195 (+ 13.4 %)
LM long queries	0.6100	0.4460	0.2800	0.3487 (+ 21.1 %)
BM25 short queries	0.4940	0.3620	0.2410	0.2595 (+ 8.6 %)
BM25 medium queries	0.5520	0.4060	0.2620	0.2992 (+ 14.7 %)
BM25 long queries	0.5980	0.4373	0.2810	0.3230 (+ 23.4 %)

Table 3 new relevance feedback algorithms

Pair wise comparison of the new relevance feedback and the ad-hoc experiments shows the same picture for the new relevance feedback algorithms as for the simple relevance feedback algorithms: The LM-algorithm shows a significant improvement in retrieval performance, but the BM25 algorithms improvement after relevance feedback is not significant according to the sign test. Pair wise comparison of the performance of the simple and the new relevance feedback algorithms shows no significant differences whatsoever.

5 Conclusion

We introduced a relevance feedback algorithm for the language model-based retrieval systems that shows a similar gain in retrieval performance as the relevance feedback algorithm of the traditional probabilistic model. The paper presented several reasons why simple approaches to relevance feedback are inappropriate for best match retrieval algorithms. However, we were unable to show that the well-motivated algorithms perform significantly better than the simple algorithms. Apparently, the simple algorithms approximate the well-motivated algorithms well enough to be realistic in real retrieval settings.

References

[1] A.P. Dempster, N.M. Laird and D.B. Rubin, Maximum likelihood from incomplete data via the EM algorithm. *Journal Royal Statistical Society 39B*, pp. 1-38, 1977.

[2] S.P. Harter, An algorithm for probabilistic indexing. *Journal of the American Society for Information Science 26*(4), pp. 280-289, 1975.

[3] D. Hiemstra, A linguistically motivated probabilistic model of IR, *Proceedings of the European Conference on Digital Libraries ECDL'98*, pp. 569-584, 1998.

[4] D. Hiemstra, Using language models for information retrieval, *Ph.D. thesis University of Twente*, 2001.

[5] D. Hiemstra, A probabilistic justification for using *tf.idf* term weighting in information retrieval, *International Journal on Digital Libraries 3*(2), pp.131-139, 2000.

[6] D. Hull and S.E. Robertson, The TREC-8 filtering track final report, *Proceedings of the 8th Text Retrieval Conference TREC-8*, pp. 35-56, 2000.

[7] W. Kraaij, R. Pohlmann and D. Hiemstra, Twenty-One at TREC-8: using language technology for information retrieval, *Proceedings of the 8th Text Retrieval Conference TREC-8*, pp. 285-300, 2000.

[8] D.R.H. Miller, T. Leek and R.M. Schwartz, A hidden Markov model information retrieval system, *Proceedings of the 22nd ACM SIGIR Conference*, pp. 214-221, 1999.

[9] K. Ng, A maximum likelihood ratio information retrieval model, *Proceedings of the 8th Text Retrieval Conference TREC-8*, pp. 285-300, 2000.

[10] J.M. Ponte and W.B. Croft, A language modelling approach to IR, *Proceedings of the 21st ACM SIGIR Conference*, pp. 275-281, 1998.

[11] S.E. Robertson, The probability ranking principle in IR. *Journal of Documentation*, 33:294-304, 1977

[12] S.E. Robertson and K. Sparck-Jones, Relevance weighting of search terms, *Journal of the American Society of Information Science*, pp.129-146, 1976.

[13] S.E. Robertson and S. Walker, Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval, *Proceedings of the 17th ACM SIGIR Conference*, pp. 232-241, 1994.

[14] S.E. Robertson and S. Walker, Okapi / Keenbow at TREC-8, automatic ad hoc, filtering, VLC and interactive, Proceedings *of the 8th Text Retrieval Conference*, pp.151-162, 2000.

[15] J.J. Rocchio, Relevance feedback in information retrieval, In: G. Salton (ed.), *The Smart retrieval system: experiments in automatic document processing*, Prentice Hall, pp. 313-323, 1971.

[16] K. Sparck-Jones, S. Walker and S.E. Robertson, A probabilistic model of information retrieval: development and comparative experiments. *Information Processing & Management 36*(6), pp. 779-840, 2000.

[17] E. Voorhees, Overview of the TREC-8 conference, *Proceedings of the 8th Text Retrieval Conference TREC-8*, pp. 1-24, 2000.