Optimization of Relevance Feedback Weights

Chris Buckley*, Gerard Salton

Department of Computer Science
Cornell University
Ithaca, NY 14853-7501
chrisb@cs.cornell.edu, gs@cs.cornell.edu

Abstract

Methods for learning weights of terms using relevance information from a learning set of documents have been studied for decades in information retrieval research. The approach used here, Dynamic Feedback Optimization, starts with a good weighting scheme based upon Rocchio feedback, and then improves those weights in a dynamic fashion by testing possible changes of query weights on the learning set documents. The resulting optimized query performs 10—15% better than the original when evaluated on the test set. We discuss the constant tension between describing what a relevant document should contain, and describing what the known relevant documents do contain.

1 Introduction

In the typical information retrieval environment, including both ad-hoc interactive retrieval and document filtering based on long-term information needs, an original query is submitted to a system which then returns documents for inspection. Users then look at those retrieved documents and submit a new query based upon their original need and the returned documents. Relevance feedback is the process of automatically altering an existing query using information supplied by users about the relevance of previously retrieved documents.

Relevance feedback has been an important research topic for well over 25 years [Ide71, Roc71, RJ76, SB90]. Increased attention has been paid to relevance feedback in the past several years due to both the increased acceptance of statistical information retrieval systems which can easily use relevance feedback, and the effects of the TREC conferences [Har93, Har94, Har95]. Evaluation of relevance feedback on the small pre-TREC test collections was notoriously difficult. The TREC routing environment offers a straightforward context in which relevance feedback can be evaluated and compared.

*This study was supported in part by the National Science Foundation under grant IRI 93-00124

SIGIR'95 Seattle WA USA© 1995 ACM 0-89791-714-6/95/07.$3.50

Recent work of our group at Cornell and others has centered on expanding the original query by large numbers of terms which occur in the relevant documents [BSA93, BAS94, BSAS95, BSA94, HC93, BCCN95]. We've shown that effectiveness continues to increase even after adding hundreds of terms using Rocchio's weighting approach [Roc71]. Robertson et al. at City University [RWJ+95] in TREC 3 showed very nice improvements by being much more selective about which terms to add, but possibly considering large number of terms. This was done by adding terms to the query only if the terms improved retrieval effectiveness on the learning set of documents.

The approach of Dynamic Feedback Optimization is similar to City's in that the query is optimized based upon retrieval effectiveness on the learning set of documents. But while City used the process to select query terms, our main goal here is to optimize the query weights of the already selected terms.

The modified Rocchio weighting formula used in our experiments is

$$\begin{aligned} Q_i^{\text{new}} &= \alpha\, Q_i^{\text{old}} \\ &+ \beta \frac{1}{|\text{rel docs}|} \sum_{\text{rel docs}} wt_i \\ &- \gamma \frac{1}{|\text{nonrel docs}|} \sum_{\text{nonrel docs}} wt_i \end{aligned}$$

Query terms with overall negative weights are dropped.

The set of terms occurring in the learning set relevant documents for the query is ranked by the number of relevant documents in which each term occurs. Those terms occurring most often are added to the original query, and all terms are reweighted using the Rocchio formula. Then small changes are made to each term weight in turn, running the changed query on the learning document set, and evaluating the result. If there is an improvement, then the changed term weight is kept, otherwise it reverts back to the weight at the beginning of the pass. This is done for each term in the query, and the process is then repeated for multiple passes over the entire query.

We can formulate four conjectures about the outcome of the proposed procedure:

- Conjecture 1: Retrieval should be improved on the test set of documents as long as the changes to each term are kept reasonably small.

Learning good weights for terms based upon a learning set of documents in general should improve performance of a query. However, it is very possible to tailor a query to the particular relevant documents in the learning set. Those groups investigating probabilistic dependency models in the early 1980's discovered this (e.g., see [YSB83]). Retrospective experiments that tested the weights using the same collection also used to generate the weights showed large improvements. The dependency models had great power in describing what a seen relevant document looked like. But they had much less predictive power to retrieve unseen relevant documents effectively. Thus it is important not to overfit the query to the learning documents. Starting with a good quality query, such as a Rocchio weighted query, and allowing only restrained changes, should ameliorate the overfitting problem.

- Conjecture 2. The original query should become less and less important as the quality of the weights of the added terms increases.

Theoretically, the relevance evidence from the learning set of documents should be sufficient to determine weights for a query, independently of the original query (assuming enough relevant documents have been found). In practice, though, it has been important to maintain a high weight for original query terms. One problem has been the poor quality of the expansion terms and their weights; the terms very often have little to do with relevance. The expanded query tends to lose its focus. If most of the highly-weighted terms are good quality, then the emphasis on the original terms should be decreased to allow the full effect of the learning information to come through.

- Conjecture 3. Independent evaluation of changed weights should be better than sequential evaluation.

A sequential approach to learning weights, where the base query is potentially modified after every term, is quite dependent on term order. Some relevant documents may have different focuses, and it is possible for one focus to dominate the others if a good term for that focus occurs early in the list of query terms. That term will be highly weighted and the relevant documents with that focus will get highly ranked. Other terms which describe a different focus will tend not to get highly weighted because they will lower the rank of documents related to the initial focus, while failing to raise the rank of documents dealing with other focuses. Within a single pass over all the query terms, the effect of each term should be evaluated independently.

- Conjecture 4. For optimum performance, terms that occur in the most relevant documents should be the ones emphasized by the learning process.

The most highly weighted terms should be those that indicate relevance. In general, they should tend to be those occurring in the most relevant documents of the learning set.

2 Experimental Setup

The optimization approach is investigated here using the TREC 3 routing task. Queries 101-150 of the TREC collection were used, with the learning set being the first two subsets of TREC documents (D1D2), and the test set being the third subset (D3). Evaluation of the effect of increasing the weight of an individual term is done by keeping track of the top 200 documents from the learning set that are retrieved by the query, both with and without the increased weight of the individual term. The top 200 documents of each query version are evaluated using average recall-precision. If the average recall-precision is higher using the increased term weight, then the term is assigned the new weight, otherwise the weight of the term remains the same.

The basic Dynamic Feedback Optimization scheme is as follows:

1. Feedback query is formed using Rocchio approach and all the relevance information from the learning set. The X terms occurring most frequently in the relevant documents are added to the query.
2. (optional) For each expansion query term, determine whether including the term at all improves effectiveness over the learning set.
3. Perform 3 passes over all query terms, potentially increasing the weight of each term in each pass. In the first pass, the default value is to increase a term weight by 50%. The default values for the second and third pass are 25% and 12.5%.
4. The reformulated query from step 3 is run against the test set documents, and evaluated using average recall-precision of all documents.

The basic scheme then has the following parameters and variations to examine:

Step 1:

1. Changes of the various Rocchio parameters to form the feedback query. The major aspect of these changes are
   (a) Changing the importance of the original query terms.
   (b) Changing the amount of expansion (both single terms and phrases).
2. Choice of what is considered the original query terms and weights. Two main possibilities:
   (a) Terms occurring in the original query text.
   (b) The fully expanded query (in which case step 2 is not done).

Step 2: Either select only terms that improve effectiveness, or automatically include all expansion terms.

Step 3:

1. Choice of whether to compare evaluations of the potential increased weight against the current query (including all previous terms considered in this pass), or against the evaluation done at the start of the pass.
2. Choice of number of passes (3 by default)
3. The ratios by which to increase term weights (0.5, 0.25, 0.125 by default).

	Optim Type	Rocchio Param	InitQ	Pass Ratios	Rcl-Prec
1.	—	—	—	—	0.3028
2.	None	2.4.1	—	—	0.3534
3.	Basic	2.4.1	orig	0,0,0	0.3471
4.	Basic	2.4.1	orig	.5,.25,.125	0.3654
5.	Basic	2.16.2	orig	.5,.25,.125	0.3853
6.	Basic	2.32.2	orig	.5,.25,.125	0.3849
7.	Basic	2.64.2	orig	.5,.25,.125	0.3835
8.	Basic	2.32.4	orig	.5,.25,.125	0.3871
9.	Basic	2.32.8	orig	.5,.25,.125	0.3845

Table 1: Choice of Rocchio parameters

3 Efficiency

These steps appear to be prohibitively expensive; however, careful maintenance of appropriate partial results render the process feasible for the routing environment, and even potentially feasible for real-time relevance feedback. Adding 50 terms to queries originally averaging 69 terms gives 119 terms per query. Performing the default algorithm (basically 3 passes plus the final retrieval), takes about 50 seconds of elapsed time per query on a single processor Sparc 20. 50 seconds is very reasonable for the routing environment, where everything is being done off-line. Both the algorithm and the implementation can be optimized further, and in a real-user relevance feedback environment (short queries, small numbers of relevant documents), may well be feasible. However, this paper does not examine either the effectiveness or efficiency of the algorithm for normal relevance feedback.

4 Experimental results

Unless otherwise specified, all optimization results are based on adding 50 single terms to the original query, using 3 passes and evaluating each potential term increase by average recall-precision of the top 200 documents when run on the learning set, D1D2. The final query is then run on the test set, D3, and evaluated using average recall-precision over the entire retrieved set.

Table 1 gives results of some base case runs, plus results of altering the initial importance of the original query term weights as compared to the weights derived from the relevant and non-relevant documents. Rocchio parameters of 2.4.1 have previously proved useful for routing in the TREC environment. These parameter values specify that the query weights should be twice the old query weight, plus four times the average weight in the relevant documents minus the average weight in the non-relevant documents. Compared with those parameters, 2.32.4, for example, would emphasize the contribution of the relevant documents by an additional factor of 8.

Run 1 is a base case run of original query using no relevance information. Run 2 uses the Rocchio relevance feedback algorithm but without optimization. Runs 3-9 are basic optimization runs with varied Rocchio parameters, where each expansion term (at its Rocchio weight) is provisionally added to the initial original query depending on whether it improves effectiveness. Then, except for run 3, three rounds of reweighting are performed for all included query terms. The first round provisionally increases the weight of a term by 50%, if that increased weight improves performance. The second and third rounds possibly increase the weight of a term by 25% and 12.5% respectively.

	Optim Type	Rocchio Param	InitQ	Pass Ratios	Rcl-Prec
8.	Basic	2.32.4	orig	.5,.25,.125	0.3871
10.	Basic	2.32.4	exp	.5,.25,.125	0.3896

Table 2: Choice of original query

	Optim Type	Rocchio Param	InitQ	Pass Ratios	Rcl-Prec
10.	Basic	2.32.4	exp	.5,.25,.125	0.3896
11.	Basic	2.32.4	exp	.5,.5,.5	0.3908
12.	Basic	2.32.4	exp	1,.5,.3	0.3911
13.	Basic	2.48.4	exp	1,.5,.3	0.3888
14.	Basic	2.64.8	exp	1,.5,.3	0.3962
15.	Basic	2.48.4	exp	1,1,1	0.3826
16.	Basic	2.64.8	exp	1,1,1	0.3868

Table 3: Choice of pass-ratios

All optimization runs show a substantial improvement over the base Rocchio Run 2, ranging up to a 10% improvement. Runs 4-7 show that with a proper emphasis on good expansion terms, the importance of the original query weight can diminish. Runs 6,8, and 9 show that the importance of the weights in the non-relevant documents remains relatively insignificant. The poor performance of Run 3 shows that the overall improvement is coming through the reweighting of terms, rather than through the choice of terms.

Given the results of Run 3 in Table 1 as compared to Run 2, perhaps the initial selection of expansion terms should be dropped, and the emphasis placed entirely on reweighting. Table 2 shows the effect of starting with all the expansion terms. There is mild improvement that is very consistent across other runs that are not included here. The remaining runs in this paper all start with the expanded query.

The last basic parameters to explore are the pass-ratio parameters, which indicate by how much a term's weight should be increased on each pass. The default parameters of 50%, 25%, and 12.5% yield a maximum total increase of just over a factor of 2. Runs 12-14 in Table 3 give a maximum possible increase of a factor of 3.9, and Runs 15 and 16 give a maximum increase of factor of 8. Despite this large possible difference between runs, all of them give results very close to each other. The major effect is that with good pass-ratios, the importance of the original query can be further diminished.

In the basic algorithm used above, the query is possibly changed after considering every term. Whether or not a term increase is beneficial may depend on the ordering of terms within a pass. If a term reweighting has a positive effect on some documents, but a negative effect on documents that were re-ranked higher by other terms within the same pass, then the term reweighting may be rejected. This effect was observed on the small collections used to debug the optimization algorithm. The "Pass" approach used in runs 17 and 18 compares each term reweighting evaluation against the evaluation at the beginning of the pass and keeps track of whether a term reweighting is successful. The query itself is only reweighted at the end of a pass, when all successful

	Optim Type	Rocchio Param	InitQ	Pass Ratios	Rel-Prec
13.	Basic	2.48.4	exp	1,.5,.3	0.3888
17.	Pass	2.48.4	exp	1,.5,.3	0.3864
14.	Basic	2.64.8	exp	1,.5,.3	0.3962
18.	Pass	2.64.8	exp	1,.5,.3	0.3931

Table 4: Commit after pass vs Commit after term

terms are changed.

However, Table 4 shows that this is not necessary for TREC. This seems somewhat counter-intuitive, but there are two possible contributing factors for it. One is the size of the TREC queries as compared to more normal queries. There will be several terms associated with each focus of a query; thus it is unlikely that a single term's focus should have much of an effect. This may explain why there is no increase in performance when the "Pass" approach is used. The decrease may be explained by the particular ordering of the expansion terms that was used. The expansion terms were ranked by decreasing occurrences in relevant documents. Thus the early ranked terms tend to be those terms more highly associated with all relevant documents, rather than those with a particular focus. As will be discussed later in the paper, a preference for terms with a general focus needs to be encouraged for good retrieval. The remaining runs in this paper do not use the "Pass" approach.

All of the runs in Tables 1-4 used a maximum of three optimization passes; Table 5 gives some results when six passes are used. With a good choice of pass-ratios, there is a very mild improvement using more passes. With a poor choice of pass-ratios, effectiveness decreases. The learning set recall-precision column provides the results of running the same queries, but run retrospectively on the learning set (and evaluated after 200 documents). The best run using learning set evaluation is the poorest run when evaluated on the test set. In Run 19, the queries have become overly optimized for the learning set, with the optimization emphasizing comparatively rare terms that happen to occur in the learning set relevant documents. This improves the retrospective effectiveness, but degrades the predictive effectiveness when the query is evaluated on previously unseen documents.

Table 6 gives some details of the weight changes for each pass on run 20. This is a basic run, starting with a query expanded by 50 terms, weighted with Rocchio parameters 2.48.4, per term evaluation done after 200 top documents on the learning set. Each pass has a pass-ratio of 1.0; ie, a term's weight is either doubled or left alone on each pass. Thus a term that is reweighted every pass would have a final weight of 64 times its original weight. The second column gives the average precision on the top 200 documents of the learning set (this is what is used to optimize the weights) for each pass. The third column gives the average precision if the query were run on the test set after that pass. The rest of the table gives the number of terms that were reweighted on that pass. This is given for the original query terms, and for buckets of 10 expansion terms. That is, the expansion term increment of 1-10 describes the number of terms reweighted (averaged over the 50 queries) among the 10 terms which occurred in the highest number relevant documents for each query.

The learning set recall-precision figures increase dramatically on the first pass and then continue to improve with the increases gradually diminishing. Note that the final value of .458 is an extremely high value for TREC average precision after 200 documents. This is a retrospective evaluation on the learning set which uses the same document set for both optimization and evaluation.

The test set recall-precision figures, though, don't match the increases of the learning set figures. There is the initial improvement on the first pass, and then the results level off and then tail off. This is not what one would have hoped given the learning set figures.

One explanation for this is suggested by the reweighting statistics in the rest of the table. There is no emphasis on reweighting those terms with the highest evidence of relevance. Indeed, by pass 6 an expansion term from the bucket with the least evidence (41-50) is three times more likely to have its weight doubled than an expansion term from the bucket with the most evidence. As the number of passes increases, the query becomes more dominated by terms which occur in comparatively few relevant documents (those expansion terms added at ranks 41-50). Some of these terms have no semantic connection with relevance, but just happen to randomly occur in more relevant documents than would normally be expected. Increasing the weight on one of these terms will improve the ranks of the relevant documents with the term, but will not bring other non-relevant documents into top ranks since those documents will tend to match only this one random term. Thus the learning set recall-precision will improve, but this improvement will not be reflected in the test set evaluation.

Each individual term weight optimization so far has been evaluated using the top 200 retrieved learning set documents. 200 was chosen to give good effectiveness at a reasonable efficiency. Table 7 gives the results using some other evaluation points. Using the top 400 helps noticeably, but increasing beyond that point yields no important improvement. Using 1000 is about twice as slow as using 200.

Table 8 gives the effect of adding more terms. Run 27 is the pure Rocchio feedback run with no optimization that was done for TREC 3. It is interesting to note that adding only 50 terms with optimized weights performs 6% better than adding the 330 terms of Run 27 with no optimized weights. As in previous experiments based on Rocchio feedback, adding more terms increases effectiveness up to a certain point. In previous experiments, effectiveness started to level off after adding 200 terms; in this experiment there is a slight deterioration between 210 and 330 documents. If effectiveness is the objective, the parameter settings of Run 25 should be used.

One worry in an experiment with this many parameters and this many test runs is that the parameters may have gotten tuned to produce good results on the test set, but not give good results in general. In this experiment, that might be happening if we were "learning" the terms contained in the test set relevant documents. To ensure that this is not happening, the final parameters are run with a new set of queries (and relevant documents). Table 9 gives the results of running with the final parameters of Run 24, but using queries 51—100 of TREC. This is the TREC 2 routing task. As can be seen from Run 28 and Run 29, Dynamic Feedback Optimization gives a 12% improvement over the standard Rocchio run for these queries as well.

5 General Discussion and Future Work

A constant theme in the above discussion is the need to optimize weights to describe relevance as opposed to opti-

	Optim Type	Rocchio Param	Pass Ratios	Rel-prec learning	Rel-prec test
19.	Basic	2.48.4	6,5,4,3,2,1	0.4692	0.3512
20.	Basic	2.48.4	1,1,1,1,1,1	0.4579	0.3730
21.	Basic	2.64.8	1,.5,.3,.2,.1,.05	0.4124	0.3981

Table 5: Increasing number of passes

Pass	Learn RP(200)	Test RP	Orig Incr	Expansion Term Incr 1-10	11-20	21-30	31-40	41-50
Init	.287	.367	69	10	10	10	10	10
1	.363	.385	25.1	3.1	3.1	3.0	2.8	3.2
2	.405	.386	17.7	2.1	2.3	2.1	2.3	2.1
3	.430	.383	13.0	1.0	1.3	1.7	1.5	1.6
4	.441	.379	8.9	0.6	1.1	0.9	1.2	1.2
5	.450	.376	6.5	0.3	0.6	0.7	0.8	0.9
6	.458	.373	4.6	0.3	0.5	0.8	0.8	0.9

Table 6: By-pass analysis for pass-ratio of 1.0 (Run 20)

mizing weights to describe the known relevant documents. If the latter is the goal, then it is easy to achieve near-perfect retrospective retrieval effectiveness. But this retrospective effectiveness does not translate into effectiveness in retrieving new documents. An example of this would be a Boolean query consisting of the "OR" of a set of document descriptions of the known relevant documents, with each document description being the "AND" of all terms occurring in the document. This query will have perfect retrospective effectiveness, but will be practically useless in retrieving new documents.

Rocchio Feedback Optimization attempts to ensure that the optimized query remains focused on the query information need rather than the details of the particular relevant documents used for learning. A good starting feedback query is used; one that has been proven to work well in many environments. That query is then modified with strict limits placed on the maximum changes that can occur. If these limits do not exist, then the results of Run 19 show the predictive effectiveness can decrease significantly.

Revisiting the conjectures:

Conjecture 1: Retrieval should be improved on the test set of documents as long as the changes to each term are kept reasonably small.

This has been shown to be true. The optimized Run 22 gives 15% improvement over original Rocchio Run 2 (both runs expanding by 50 terms). Run 24 gives an 11% improvement over Run 26; Run 26 being the approach used for the Cornell TREC 3 routing submission.

Conjecture 2. The original query should become less and less important as the quality of the weights of the added terms increases.

This has also been verified. In the final formulation, the ratio of relevant document importance to original query importance has increased by a factor of 16 over the original ratio with unoptimized weights.

Conjecture 3. Independent evaluation of changed weights should be better than sequential evaluation.

This does not seem to be true in this experiment. One explanation may be that the particular ordering of expansion terms we used tends to preserve the focus of the query. Terms that appear in more relevant documents are acted upon first, so the narrower terms do not have a chance to shut them out.

Conjecture 4. For optimum performance, terms that occur in the most relevant documents should be the ones emphasized by the learning process.

This also remains unverified. Our experiments showed that terms occurring in few relevant documents had even more chance of being emphasized in the optimization than terms in the original query, or terms that occurred in the greatest number of relevant documents. It remains for further study to see whether this emphasis is correct or whether it is due to improper tuning of the query to fit particular documents in the learning set. One approach for examining this would be to weight the pass-ratio by the amount of evidence for each term. For example,

$$\text{PassRatio}(t_i) = \text{OrigPassRatio} * \frac{\text{NumRel}(t_i)}{\text{TotNumRel}}$$

One slightly unexpected result was that selection of expansion terms did not work. We used roughly the same criteria as City University did in TREC 3 in deciding whether a term should be added or not: a term is included if and only if including it improves effectiveness on the learning set. City got large improvements using this as a filter; we get better results if we include all terms. Work needs to be done reconciling these results.

One general area that needs attention is document length normalization. The Rocchio feedback weight is heavily dependent on the weights in the relevant documents, which in turn are heavily dependent on the cosine length normalization. The cosine normalization technique was developed in an environment where the documents have a single focus. It is not as applicable to the TREC environment of longer full-text documents which discuss several topics. Either we need to concentrate on using passages where the passage has a single focus, or we need to change the normalization technique. At the moment, the weights in long relevant documents tend to be much lower than the weights in short relevant documents, and thus do not contribute as much as they should to the final query weight.

In conclusion, Dynamic Feedback Optimization works well, giving a 10-15% improvement over standard Rocchio

	Eval Thresh	Rocchio Param	Pass Ratios	Rcl-Prec
21.	200	2.64.8	1,.5,.3,.2,.1,.05	0.3981
22.	400	2.64.8	1,.5,.3,.2,.1,.05	0.4065
23.	1000	2.64.8	1,.5,.3,.2,.1,.05	0.4078

Table 7: Increasing number of top documents used in evaluation

	Expansion (term.phrase)	Rocchio Param	Pass Ratios	Rcl-Prec
22.	50.0	2.64.8	1,.5,.3,.2,.1,.05	0.4065
24.	100.10	2.64.8	1,.5,.3,.2,.1,.05	0.4144
25.	200.10	2.64.8	1,.5,.3,.2,.1,.05	0.4228
26.	300.30	2.64.8	1,.5,.3,.2,.1,.05	0.4205
27.	300.30	2.4.1	—	0.3825

Table 8: Increasing number of expansion terms

feedback. The effectiveness is dependent on optimizing enough to improve effectiveness, but not so much as to be describing the learning set relevant documents instead of describing the information need of the user. This is done by starting with a good query, and limiting the changes due to the optimization procedure. Open questions and conjectures remain that need to be examined in future work.

References

[BAS94] Chris Buckley, James Allan, and Gerard Salton. Automatic routing and ad-hoc retrieval using SMART : TREC 2. In D. K. Harman, editor, *Proceedings of the Second Text REtrieval Conference (TREC-2)*, pages 45–56. NIST Special Publication 500-215, March 1994.

[BCCN95] John Broglio, James Callan, W. Bruce Croft, and Dan Nachbar. Document retrieval and routing using the INQUERY system. In D. K. Harman, editor, *Proceedings of the Third Text REtrieval Conference (TREC-3)*. NIST (in preparation), 1995.

[BSA93] Chris Buckley, Gerard Salton, and James Allan. Automatic retrieval with locality information using SMART. In D. K. Harman, editor, *Proceedings of the First Text REtrieval Conference (TREC-1)*, pages 59–72. NIST Special Publication 500-207, March 1993.

[BSA94] Chris Buckley, Gerard Salton, and James Allan. The effect of adding relevance information in a relevance feedback environment. In W. Bruce Croft and C.J. van Rijsbergen, editors, *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 292–300, New York, July 1994. Springer-Verlag.

[BSAS95] Chris Buckley, Gerard Salton, James Allan, and Amit Singhal. Automatic query expansion using SMART : TREC 3. In D. K. Harman, editor, *Proceedings of the Third Text REtrieval Conference (TREC-3)*. NIST (in preparation), 1995.

	Expansion (term.phrase)	Rocchio Param	Pass Ratios	Rcl-Prec
28.	200.10	2.64.8	1,.5,.3,.2,.1,.05	0.4542
29.	300.30	2.4.1	—	0.4045

Table 9: TREC-2 Routing (Queries 51-100)

[Har93] D. K. Harman. Overview of the first Text REtrieval Conference (TREC-1). In D. K. Harman, editor, *Proceedings of the First Text REtrieval Conference (TREC-1)*, pages 1–20. NIST Special Publication 500-207, March 1993.

[Har94] D. K. Harman. Overview of the second Text REtrieval Conference (TREC-2). In D. K. Harman, editor, *Proceedings of the Second Text REtrieval Conference (TREC-2)*, pages 1–20. NIST Special Publication 500-215, March 1994.

[Har95] D. K. Harman. Overview of the third Text REtrieval Conference (TREC-3). In D. K. Harman, editor, *Proceedings of the Third Text REtrieval Conference (TREC-3)*. NIST, in preparation, 1995.

[HC93] David Haines and W. Bruce Croft. Relevance feedback and inference networks. In Robert Korfhage, Edie Rasmussen, and Peter Willett, editors, *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2–11, New York, 1993. Association for Computer Machinery.

[Ide71] E. Ide. New experiments in relevance feedback. In Gerard Salton, editor, *The SMART Retrieval System—Experiments in Automatic Document Processing*, chapter 16. Prentice Hall, Englewood Cliffs, NJ, 1971.

[RJ76] S.E. Robertson and K. Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27(3):129–146, May-June 1976.

[Roc71] J.J. Rocchio. Relevance feedback in information retrieval. In Gerard Salton, editor, *The SMART Retrieval System—Experiments in Automatic Document Processing*, chapter 14. Prentice Hall, Englewood Cliffs, NJ, 1971.

[RWJ+95] S Robertson, S Walker, S Jones, M M Hancock-Beaulieu, and M Gatford. Okapi at TREC 3. In D. K. Harman, editor, *Proceedings of the Third Text REtrieval Conference (TREC-3)*. NIST (in preparation), 1995.

[SB90] Gerard Salton and Chris Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41(4):288–297, 1990.

[YSB83] Clement Yu, Gerard Salton, and Chris Buckley. An evaluation of term dependence models in information retrieval. In G. Salton and H.J. Schneider, editors, *Research and Development in Information Retrieval,Lecture Notes in Computer Science 146*, pages 151–173. Springer-Verlag, Berlin, 1983.