A Language Modeling Framework for Selective Query Expansion

Steve Cronen-Townsend Yun Zhou W. Bruce Croft

{crotown, yzhou, croft}@cs.umass.edu
Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts, Amherst, MA 01003

ABSTRACT

Query expansion is a well-known technique that has been shown to improve *average* retrieval performance. This technique has not been used in many operational systems because of the fact that it can greatly degrade the performance of some individual queries. We show how comparison between language models of the unexpanded and expanded retrieval results can be used to predict when the expanded retrieval has strayed from the original sense of the query. In these cases, the unexpanded results are used while the expanded results are used in the remaining cases (where such straying is not detected). We evaluate this method and others on a wide variety of TREC collections and show how to automatically compute a decision threshold for a collection. We demonstrate the ability of the method to enhance the effectiveness and reliability of the query expansion technique in information retrieval.

Keywords
language models, query expansion

1. INTRODUCTION

When examining the results of any query expansion method over a large number of queries, one always finds that nearly equal numbers of queries are helped and hurt by the technique. A good technique for a given type of query and collection helps, on average, slightly more than it hurts. Thus the best simple strategy to improve the average performance is often to treat every query the same and expand every one. This strategy has the unfortunate effect of making the results for certain queries much worse than for retrieval without expansion.

In this paper we develop a method for discriminating between queries and deciding when not to use the results of an expansion technique that is likely to hurt the retrieval performance *for that particular query*. We explore our method in a language modeling framework where ordinary retrieval is done by the query likelihood method[18] and expanded retrieval is done using relevance models[15, 8]. Our best method uses only the documents and their ranks in the ranked list to form language models. So, since our method compares models of each *ranked list* of documents it can be applied to any choice between two retrieval techniques, whether they are based on language modeling or not. One of the ranked lists could be influenced by user feedback, for example.

These techniques are important because they are the first reliable steps at solving the crucial problem of consistency in applying query expansion to real systems. Thus the correct measure of these techniques is not in their admittedly small effect on global retrieval effectiveness. Although they have some success at detection of bad-to-expand queries, this is only part of their significance. Most importantly they provide an extensible framework that makes the first steps toward solving a previously unassailable, but very important, problem.

First we examine the unexpanded and expanded retrieval methods we use. In the next section we introduce three methods to predict queries for which expanded retrieval performs poorly relative to unexpanded retrieval and compare the methods. We go on to examine how to set a decision threshold for the best method, model comparison, to make the method applicable to real collections of documents. Then we discuss related work, future work on our methods, and conclusions.

2. RETRIEVAL

The first step in predicting poor expansion results is performing expanded and unexpanded retrievals themselves. The difference in the average precision of these retrievals (expanded minus unexpanded) becomes the figure-of-merit of most interest to us for developing and testing methods. We term this average precision change the *improvement* of the query on expansion, and we seek to predict when it will be significantly negative. By working with TREC test collections, where the improvement can be calculated, we seek to develop a method robust enough to be applied confidently to new collections.

In our case, we adopt a language modeling framework[9] where the simplest retrieval technique, query likelihood, is

the obvious choice for our unexpanded (baseline) result for each query. For our expanded retrieval results for each query, we use relevance model retrieval[15, 8]. Relevance model retrieval is a conceptually simple, principled, and effective expansion technique that fits cleanly into our language modeling framework. It ranks the documents by the closeness of their language models to a mixture of models of top documents for the query, where the mixing weight for each document model is its likelihood of generating the query.

2.1 System details

Our collections are indexed in standard Lemur fashion[16] with upper case and punctuation ignored. Single characters and digits and terms on the InQuery stop list[3] are removed and Krovetz stemming[13] is applied.

Throughout this we work we use a language modeling framework. In our case, a *language model* refers to an estimated probability distribution over vocabulary terms, approximated as occurring independently. We utilize two kinds of language models, *document models* of the language usage in individual documents, and a *collection model* where the probability estimate for each term is simply taken to be the relative frequency of the term in the entire collection of documents.

2.2 Unexpanded retrieval

For unexpanded retrievals, we use query likelihood[18] where each document is scored by the likelihood of its model generating the query.

To calculate this score, we construct a smoothed unigram language model, $P(w|D)$, for each document, D, which is simply a probability distribution over term occurrences. Here w represents any term, but we only need to compute the probability estimates, $P(w|D)$, for w being a query term for this calculation. We perform a query likelihood retrieval by scoring each collection document, D, by

$$P(Q|D) = \prod_{q \in Q} P(q|D), \tag{1}$$

where Q is the query and q is a query term[18]. $P(Q|D)$ is an estimate of the likelihood of a document's model generating the query terms under an independence approximation.

In constructing the document model probability estimates for query terms, we found it important to use Dirichlet smoothing[21] rather than the linear smoothing originally used to compute clarity scores[11]. In particular,

$$P(w|D) = \lambda P_{ML}(w|D) + (1 - \lambda) P_{ML}(w|\text{coll}), \tag{2}$$

where $P_{ML}(w|D)$ is simply the number of times w occurs in document D divided by the number of term occurrences in D. $P_{ML}(w|\text{coll})$ is an identical estimate for the entire collection. For Dirichlet smoothing λ is different for each document, with

$$\lambda = \frac{\|D\|}{\|D\| + \mu}, \tag{3}$$

where $\|D\|$ is the number of term occurrences in D and μ is a constant. Substitution of Equation (3) into Equation (2) and simplification yields the usual expression for Dirichlet-smoothed estimates given in [21].

We used a consistent Dirichlet prior of $\mu = 1000$ across all collections, for simplicity. This choice gave reasonable retrieval performance across collections and allowed us to focus on the prediction task, rather than tuning the retrieval task in a way that would not be possible with a new collection without test queries and relevance information.

2.3 Expanded retrieval

The key to expanded retrieval with a relevance model is estimating the relevance model itself. Each collection document is then scored for retrieval by the closeness of its model to this relevance model. From the query expansion point of view, the relevance model is the expanded query that has a probability (weight) for every term in the vocabulary[15].

2.3.1 Relevance models

The first step in estimating a relevance model was already done for the unexpanded retrieval; it is the scoring of documents with Equation (1). We use the identical scores described above and used for the unexpanded retrieval. From $P(Q|D)$ we obtain $P(D|Q)$ by Bayes Rule:

$$P(D|Q) \propto P(Q|D)P(D). \tag{4}$$

Since $P(Q)$ does not depend on D the proportionality holds. With uniform prior probabilities, $P(D)$, for the documents this amounts to a normalization since we require $P(D|Q)$ to sum to 1 over documents.

The second step actually estimates the relevance model, $P(w|Q)$, as a weighted average of document models, with the estimates of $P(D|Q)$ serving as mixing weights:

$$P(w|Q) = \sum_{D \in R} P(w|D)P(D|Q). \tag{5}$$

The documents models that are mixed are linearly smoothed with $\lambda = 0.9$ in Equation 2. Using the $P(D|Q)$ as mixing weights ensures that documents containing as many of the query terms and frequently dominate the average. Models of the top 50 documents are mixed with Equation 5 and the models are truncated to the top 1000 terms for efficiency.

2.3.2 Relevance model retrieval

The actual retrieval is a third step where each document is scored by the closeness (judged with cross-entropy) of a its model to the relevance model constructed in the first two steps. Here the document models over all terms are estimated using linear smoothing with $\lambda = 0.2$ in Eq. 2.

Each of the three steps in relevance model retrieval requires individualized smoothing for good performance[14]. The choices of smoothing types (Dirichlet, linear, and linear, respectively) and parameters given above were made on the basis of reasonable mean average precision across different collections. We kept the parameters constant across collections to keep the emphasis on application of our methods to new collections where relevance information is not available to tune the parameters. But since we did not tune the parameters for each collection, our results are not strictly comparable to papers on retrieval methods where many parameters are tuned for each test collection individually for best retrieval performance. We state all the parameter settings we use, in order to allow reproduction of our results.

3. EXPANSION PREDICTION TASK

We now introduce three different methods we have tried to predict queries that perform poorly when expanded. The

three methods are the *clarity method*, the *overlap method*, and the *model comparison method*. The first method uses the clarity score of a query as a potential predictor, the second uses the overlap in the document IDs that occur in the top ranks of expanded and unexpanded ranked lists, and the third, and most successful, uses a direct comparison between models of the unexpanded and expanded ranked lists.

3.1 Clarity method

Our simplest idea for expansion prediction is to base it on the clarity measure used for predicting the average precision of document retrieval[11]. If a query has a high clarity score in a collection (predicting an effective query likelihood retrieval), perhaps expanding it is unwise, on average. This idea is the basis of the clarity method.

To calculate a clarity score for a query in a collection we estimate a relevance model for the query in the collection, as described previously. The same relevance model computed for relevance retrieval (see Section 2.3.1) is used for clarity score calculation in our implementation. A query is scored by how different its relevance model is from an overall average model of the collection. We now describe clarity scores in more detail.

3.1.1 Weighted clarity scores

The clarity score was originally defined[10] as the relative entropy (also known as the Kullback-Liebler(KL) divergence[7]) between a query's relevance model in the collection, $P(w|Q)$, and a model of the entire collection $P(w|\text{coll})$ by

$$\text{clarity} = D(P(w|Q)||P(w|\text{coll})) \tag{6}$$

For the collection model in this study we mean a language model with a relative frequency in the entire collection used to estimate the probability of each term.

We extend clarity scores by using the weighted relative entropy[19, 1]

$$D(A||B;U) = \frac{1}{E(A;U)} \sum_{events,i} u_i a_i \log_2 \frac{a_i}{b_i}, \tag{7}$$

where A and B represent probability distributions and U represents a vector of weights over events. The normalization factor $E(A;U) = \sum_j a_j u_j$, where a_i and b_i represents the probability of event i according to the A and B distributions, respectively. The weighted relative entropy is the expectation value of the quantity $Log_2 \frac{A}{B}$ using a weighted version of the A distribution instead of the plain A distribution as in standard relative entropy(KL).

In the cases we present, an event is the occurrence of a term. A generalized clarity measure is then defined by the weighted relative entropy from the relevance model, $P(w|Q)$, to the collection model $P(w|\text{coll})$

$$\text{clarity} = \sum_{w \in V} \frac{u(w)P(w|Q)}{\sum_{w' \in V} u(w')P(w'|Q)} \log_2 \frac{P(w|Q)}{P(w|\text{coll})}, \tag{8}$$

where $u(w)$ are the term weights and V is the vocabulary of the collection. When $u(w) \equiv 1$ for all terms in the vocabulary this expression is just the usual Kullback-Liebler(KL) divergence[7] and leads to the clarity scores used previously[11]. The ordinary KL divergence is the expectation value in the first (A)distribution of the difference in the log probabilities of an event in the two distributions (since $log_2 \frac{a}{b} = \log_2 a - \log_2 b$. The weighted relative entropy is an expectation value of the same quantity with a *weighted* version of the first distribution being used to calculate the expectation value.

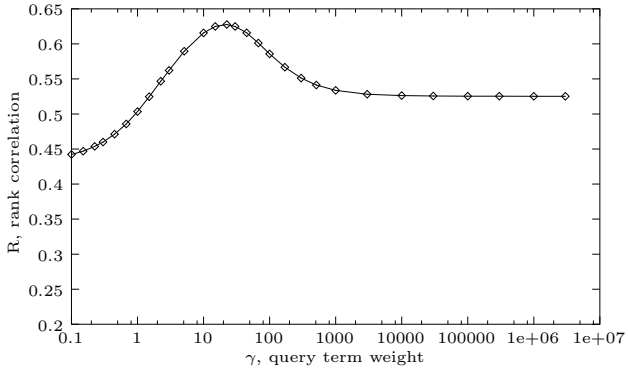

Figure 1: The rank correlation between clarity score and query likelihood document retrieval average precision as a function of the query term weight, γ, for the 1804 queries of the TREC 9 Query Track aggregate against TREC Disk 1.

We became interested in this formulation of clarity scores as we realized that differences in the probability estimates for some terms are more meaningful than others. This difference can be described in the weight or utility vector $u(w)$.

The most important term information in information retrieval is whether or not each vocabulary term occurs in the query. To reflect this, we tried giving each query term weight $u(q) = \gamma$ and all other terms weight $u(w) = 1$. Here γ represents how many times more significant the occurrence of a query term is than the occurrence of another term in a document. We also tried giving each query term, q, weight $n(q)\gamma$ where $n(q)$ is the number of times the given term appears in the query. Even in test collections such as the TREC 9 QueryTrack aggregate where query terms do repeat on occasion, the results of the two schemes are nearly identical. With either implementation, the identical large improvement is seen in the rank correlation between the clarity scores and the average precisions of the queries.

As shown in Figure 1 for the TREC 9 Query Track, the clarity score predicts average precision better as the weight of query terms in measuring the degree difference from the collection model is increased. The value $R = 0.50$ at $\gamma = 1$ (unweighted) is significantly higher than the value of $R = 0.39$ reported for the original clarity score method for the Query Track aggregate[11]. This difference is due to our use of Dirichlet smoothing, rather than linear smoothing, for the scoring of documents by query likelihood. As the relative importance of query terms to the comparison is increased, the correlation rises to a maximum of $R = 0.62$ at $\gamma = 30$.

At $\gamma = \infty$, the clarity score is computed over just the terms that appear in the query. This setting gives more correlation with retrieval performance ($R = 0.53$) than using the entire relevance model for comparison ($R = 0.50$). Thus, in applications requiring retrieval precision prediction where a full relevance model is not needed, a *reduced relevance model* may be computed solely for the purpose of clarity score computation. For these reduced relevance models the

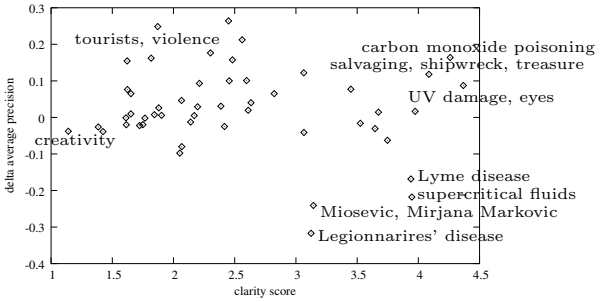

Figure 2: A scatter plot of Δ average precision versus the clarity score for each of the 50 queries of TREC 8. The Δ average precision is the value with relevance retrieval(Rel) minus the value with query likelihood retrieval(QL). The clarity scores are calculated as described in Section 3.1.1, without weighting, on models formed as described in Section 2.3.1.

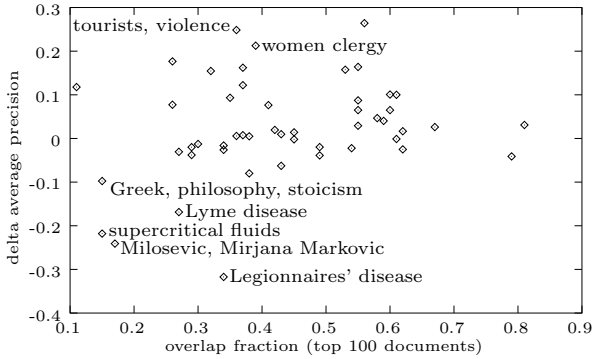

Figure 3: A scatter plot of Δ average precision and the overlap score for each of the 50 queries of TREC 8. The Δ average precision is the value with relevance retrieval(Rel) minus the value with query likelihood retrieval(QL). The overlap score is the fraction of the top 100 documents in the unexpanded ranked list that are also in the top 100 of the expanded ranked list. The same 7 queries are labelled as in Figure 4.

probabilities only need be estimated for the query terms themselves. This calculation can be done with just the lists of documents containing each query term and the number of term occurrences in each. Since this information is typically stored in indices for information retrieval, this computation can be made extremely efficient.

The optimal value of γ for predicting retrieval performance does not vary much over the collections we tested. Very good performance is always found for $\gamma \approx 30$.

Initial tests with the original formulation of clarity scores [10, 11] showed little relationship between clarity score and expansion gains, but we found that using Dirichlet smoothing to score the documents improved the predictive ability of the relevance models for predicting expansion improvements as well as predicting document retrieval performance.

3.1.2 Results

The clarity method has the advantage that it does not require doing the expanded retrieval to calculate the score. Bad-to-expand queries have a high score quite reliably, as well, as seen in Figure 2. Unfortunately, good-to-expand queries also rather frequently have high scores. For example, The query "salvaging, shipwreck, treasure" which is good-to-expand has almost the same high clarity score as "supercritical fluids," which is bad-to-expand. Since we know no way of distinguishing the two kinds of high-scoring queries, automatic thresholding of this score seems impossible. The most a system could say, given a high-scoring query, is that the query *might* behave badly on relevance model expansion. In such a case a system could save the cost of doing the expanded retrieval sometimes by showing the query likelihood retrieval results and making the expanded retrieval optional for the user. Query term weighting was not consistently helpful (as it is in retrieval prediction). The same weighted comparisons become crucial in the model comparison method, however. The effect of the clarity method on global retrieval effectiveness (using unweighted clarity scores) is seen in the fourth column of Table 1, when an estimated Bayes optimal decision threshold is used.

3.2 Overlap method

The overlap method measures the overlap of document IDs in the unexpanded and expanded ranked list. It was proposed by Chris Buckley and Ellen Voorhees while summarizing the work of the recent ARDA Northeast Regional Research Center (NRRC) workshop on Reliable Information Access (RIA) and is the only method competing with the clarity method and the model comparison method that we know of.

Though the method was a work in progress at the time we learned of it, we have attempted to reproduce it for the sake of comparison. We implement it as the simple fraction of documents in the top N documents of the unexpanded ranked list that are also present in the top N documents of the expanded ranked list. See Sections 2.2 and 2.3 for how we carry out the retrievals. For best prediction we settled on $N = 100$ for the overlap scores in this paper.

This score is low when the two retrievals do not share many documents in the top N and a low score provides evidence that expansion would hurt the results. This is the opposite of the two statistical measures (clarity and model comparison) we propose here, where a high score is evidence not to expand the query. The improvement (change in average precision on expansion) is shown against the overlap scores for each of the 50 TREC 8 queries in Figure 3.

The only complication in implementing the technique is posed by the rare queries that returned fewer than 100 results for query likelihood retrieval in Lemur[16], the information retrieval system we use for our implementations. This is possible since Lemur currently only scores documents in query likelihood retrieval that contain at least one query term. So in the case of rare query terms that only occur in a small number of collection documents it is possible to get fewer than 100 documents scored. In these cases we simply padded the lists with null documents so that there were 100 documents for the overlap calculation. This implementation should not affect the performance of the technique since it only affects a couple queries per collection and it results in consistently low overlap scores for these queries (since their max possible overlap is limited to n/N where n is the number of documents returned by the system). Even if this score

Collection	Queries	Relevance Model	Clarity	Document ID Overlap	Model Comparison top terms	Perfect Choice
AP88-90	#51-#150 title	0.2875	0.2744	flip	ident	0.3046
TREC 1+2+3	#51-#150 title	0.2490	ident	flip	ident	0.2589
TREC 5	#251-#300 title	0.1609	0.1556	0.1526	0.1644(∗)	0.1837
TREC 6	#301-#350 title	0.2013	0.2168(∗)	0.2141(∗)	0.2115(∗)	0.2468
TREC 7	#351-#400 title	0.2524	0.2229	0.2005	0.2212	0.2711
TREC 8	#401-#450 title	0.2715	0.2719(∗)	0.2692	0.2756(∗)	0.3011
Query Track Aggregate	1804 Variations: #51-#100	0.2219	0.2082	0.1707	0.2188	0.2387

Table 1: Selective mean average precision(MAP) with estimated Bayes optimum thresholds chosen using the full relevance information in each test set. The fourth line indicates that on the TREC 6 ad hoc collection with the titles of TREC topic #301-#350 as queries the performance with all relevance model retrieval is 0.2013. The "Perfect Choice" values are the upper bound MAP if the best retrieval method is chosen on a query-by-query basis knowing the performance of each method. "ident" indicates that the score distributions for positive improvement queries and negative improvement queries are nearly identical and "flipped" indicates the opposite relationship between extreme scores and performance is exhibited by the estimated score distributions. A (∗) indicates a mean average precision higher than using relevance model retrieval for every query.

is artificially low, that generally leads to the correct decision for these rare queries: do not expand because the overlap is very low.

3.2.1 Results

The overlap method has the advantage that, once the two retrievals are done, it is very simple to compute. As seen in Figure 3, bad-to-expand queries usually have low overlap scores. However, the method suffers from the same problem that the clarity method does, namely some good-to-expand queries have scores that are similar to scores of bad-to-expand queries. For example, the overlap score of the good-to-expand query, "tourists, violence," is only a little higher than the score of the bad-to-expand query "Lyme disease." Thus a low overlap score could only indicate a suspicious query, not one that would perform badly with expansion on average.

The effect of the overlap method on global retrieval effectiveness, when an estimated Bayes optimal threshold is used, is seen in the fifth column of Table 1.

This leads us to the model comparison method, which takes into account commonalities in word usage between ranked lists, rather than just commonalities in document IDs. We describe this method next.

3.3 Model Comparisons

The model comparison method, our best method for expansion prediction, compares models of the language usage in the unexpanded retrieval ranked list (model A) and in the expanded retrieval ranked list (model B). With this comparison we attempt to sense when an expanded retrieval has strayed from the original sense of a query. Comparison scores focussed on important terms in the unexpanded model are high when expanded results use these important terms much less frequently than unexpanded results. This usually indicates a poor expansion outcome (negative improvement). We next explain how we estimate ranked list language models and how we compare them.

3.3.1 Ranked List Models

A ranked list model is a mixture model using the rank of a document in a ranked list to compute its mixing weight. No probability of relevance (or similar) score according to a retrieval system is required. This freedom is necessary for the use of ranked list models in our language modeling implementation of the model comparison method, since the cross-entropy scores in relevance model retrieval are not suitable for mixing documents. Yet, they rank successfully, and this technique uses *only* the ranking to calculate the mixing weights.

Instead of estimating $P(D|Q)$ using Equation 4 we estimate it directly from the rank of the D in the ranked list for Q with $P(D|Q) = P(\text{rank of } D|Q)$, where $P(\text{rank}|Q)$ is the probability that a document at that rank is relevant (approximated as independent of Q). We then use Equation 5 to mix the documents and form the ranked list model.

For this study, we use equal probabilities for the top 100 documents in a ranked list and estimates of zero for all other documents. Thus, each ranked list model used in this study is an equal mixture of (language models of) the top 100 documents in a ranked list. The model comparison method simply compares two ranked list models, one for the unexpanded ranked list and one for the ranked list with expansion. Equal mixture models of top documents proved the best simple method, but there is clearly room to hone these models further. We used a linearly smoothed document model with $\lambda = 0.6$ to mix the ranked list models for this study.

3.3.2 Comparing ranked list models

The comparison is done with the weighted relative entropy, Equation 7, as $D(A||B;U)$. We have tried several methods for the weights U. Using equal weights over all terms (standard KL divergence) assumes differences between the models are equally meaningful for all terms in the vocabulary.

A crucial insight is that differences in the usage of all terms are *not* equally important. In particular, we want

the comparison to weight the most important terms in the unexpanded model highly.

Weighting terms based on their probabilities leads to measures with little relation to expansion performance since it compares the two models on many generic, commonly occurring terms as well as those that are genuinely important to the unexpanded model.

To pick the top terms we use the top T terms in contribution to the clarity score of the unexpanded model. We compute the clarity contributions,

$$\text{contrib}(w) = P(w) * Log_2[P(w)/P(w|\text{coll})], \qquad (9)$$

where $P(w)$ is the probability of a term w in the model and $P(w|\text{coll})$ is the probability of the term in the entire collection and take the top T terms on this score. Since these are the terms in the unexpanded model that are most unusual relative to the overall collection statistics, this forms a suitable measure of importance in the model. These top T terms are all given weight 1 and all other terms are given weight 0. There is obviously room here for possible improvement of the method, but tests show that the method is not very sensitive to T as long as it is in the range 5 to 50. By $T = 100$ the value of the score as a predictor of expansion failures suffers noticeably.

The model comparison score is an expectation of the difference in log probabilities for the important unexpanded terms in the two models of the ranked lists. Using the top T important terms, it is an average (under the A distribution, see Equation 7) of $Log_2P(w|A) - Log_2P(w|B)$ when w is each of these important terms. Thus a highly positive value indicates that the important terms in the unexpanded model are used much less frequently in the expanded retrieval. This often indicates an unsuccessful use of query expansion. In this case a system would show the user the *unexpanded* results for the query. If the score is negative, the expanded retrieval uses the important terms more than the original model, which generally indicates a good expansion result. A score of zero indicates that the two models use the important terms equally.

Figure 4 shows our model comparison scores applied to the TREC 8 ad hoc data. Highly positive comparison scores are an indicator that the performance of the expanded retrieval may be significantly worse than the unexpanded retrieval. Moreover, highly positive scores, when they occur, are well separated from the other scores. This is important because, to use the scores in a real system, the system must be able to decide how high a score constitutes justifying a decision not to use the expansion results. Additionally, we see that there are no queries in TREC 8 that are good to expand with extremely high positive scores. All these considerations are crucial to the success of the automatic thresholding we discuss in Section 4.

3.3.3 Results

Model comparison, our best method, has the great advantage that it produces scores for some queries that are generally very well separated from the scores of queries that are good-to-expand (see Figure 4). Additionally, the scores are easily interpreted and reflect the average usage difference of the most important terms in the original model.

Examining the extreme case queries in the TREC 8 scatter plot shown in Figure 4 is illuminating. Of the three highest scoring queries, all perform badly on expansion. Thus

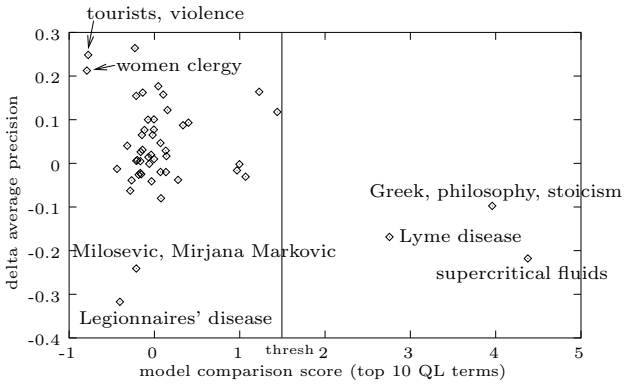

Figure 4: A scatter plot of Δ average precision and model comparison scores for each of the 50 queries of TREC 8. The Δ average precision is the value with relevance retrieval(Rel) minus the value with query likelihood retrieval(QL). The model comparison score is the KL divergence from a QL ranked list model to a Rel ranked list model computed over just the top 10 terms in clarity contribution to the QL model. The automatic threshold indicating a score greater than 95% of random one term queries (see Section 4) is shown as a vertical line labeled 'thres' and seven extreme case queries are labelled.

the model comparison leads to three correct predictions. The particular queries are "Lyme disease," "Greek, philosophy, stoicism," and "supercritical fluids." In these cases, loosening the requirement that documents use the exact query terms frequently to be highly ranked (by going from unexpanded to expanded retrieval) ranks documents too highly that do not contain all the correct jargon terms frequently. In these cases matching the exact technical terms is what satisfying the information need (TREC topic) requires. There are no other highly scoring queries to potentially be confused with these bad-to-expand queries. As far as very low scoring queries that are good-to-expand (another correct prediction case) we have the examples "tourists, violence" and "women clergy." In both these cases broadening the search to contain closely related terms can plausibly help, and the wording of the TREC topics confirms this.

For the one bad prediction case that exists: low scoring queries that are actually good to expand, we have two examples. These are the queries "Milosevic, Mirjana Markovic" and "Legionnaires' disease." The existence of queries in this region of the scatter plot means that not all bad-to-expand queries can be detected by the model comparison method. In both cases the models of expanded retrieval use the top unexpanded terms somewhat more than in the expanded model. In the case of the "Milosevic, Mirjana Markovic" query, documents must match some form of Mirjana Markovitch's name to be relevant. The model comparison method is not picking up the fact that the expanded model is losing those terms in particular because some other top terms are used more frequently in the expanded model. This suggests that query terms themselves should be weighted higher, though simply combining our query term γ weighting with this top term even weighting scheme does not help consistently. In the case of "Legionnaires' disease" the important comparison is to the correctly predicting query "Lyme

Method	score dev all queries	average score good(18)	average score bad(7)	sep
clarity	0.867	2.650	3.146	0.57
doc ID overlap	0.159	0.431	0.270	1.01
model comp: top terms	0.996	0.0656	1.470	1.41

Table 2: TREC 8 separation for the three methods: clarity, overlap, model comparison. There are 18 "good" queries with $\Delta > 0.05$ and 7 "bad" queries with $\Delta < 0.05$. The separation "sep" is the absolute value of the difference of the average scores between good and bad queries, divided by the sample standard deviation of score for all queries. For the clarity and model comparison methods, queries of high score are candidates not to expand. With the overlap method, queries with low overlap between the two ranked lists are candidates not to expand.

Method	score dev all queries	average score good(14)	average score bad(13)	sep
clarity	0.922	2.714	2.911	0.21
doc ID overlap	0.166	0.452	0.405	0.29
model comp: top terms	1.251	−0.0404	1.843	1.50

Table 3: TREC 6 separation for the three methods: clarity, overlap, and model comparison. See Table 2 for column explanations.

disease." The crucial difference is that the expanded documents can use the top terms related to the unexpanded model without being about the specific disease at all, since legionnaire is a synonym for soldiers or veterans and there are documents using those related terms that also use the term disease. The "Lyme disease" query does not have this problem since "Lyme" is a place name and has no other disease-related usages in the collection.

3.3.4 Three Methods Compared

One way to compare the three methods is to perform each with an estimated Bayes optimal threshold and to compute the mean average precision for the selective query expansion process. This produces Table 1. The thresholds are set by estimating score distributions for the positive improvement queries and negative improvement queries using heavily-smoothed kernel estimates[2] and setting the threshold where the distributions intersect[12]. The tested collection where the method shines, TRECs 5, 6, and 8, are easily seen. Further comparison of the three methods will be possible once automatic thresholding is discussed in Section 4.

Another way to compare the three methods is by quantifying the separation between the scores of good-to-expand queries and bad-to-expand queries. This is done for two collections that offer good possibility for improving over all relevance model retrieval (TREC 8 in Table 2 and TREC 6 in Table 3). With this metric, we see that the model comparison offers a strikingly higher separation between the scores of good-to-expand and bad-to-expand queries. It is this separation that makes automatic thresholding of this method a possibility.

4. THRESHOLDING

Given the great differences in the methods' separation between the scores with the opposite meanings, we only attempted automatic thresholding of the model comparison scores. The basic idea behind our thresholding scheme is to estimate the probability density for comparison scores in order to make a meaningful guess at how high a score justifies not showing expansion results. We answer the question relative to the underlying probability density for the scores.

We estimate the probability density function over scores by randomly sampling terms from the collection's vocabulary and using Gaussian kernels to smooth the results[2]. The results for TREC 8 are shown in Figure 5. The distribution estimated by computing the model comparison score for random one-term queries in the TREC 8 collection bears striking similarity to the distribution that apparently underlies the scores of real TREC test queries (compare Figure 5 and Figure 4). The automatic threshold at score 1.52 is indicated on both plots. Note that there is a high probability of scores around zero with a shoulder around score 1 and extremely low probability density for scores greater than 2. So the three queries above score 2.5 can reliably be identified as possessing unusually high model comparison scores.

After checking many policies for heuristically setting the threshold we chose to set the threshold, for any collection, so that 95% of random one-term queries have a lower model comparison score. This results in deciding not to expand queries with objectively very high model comparison scores. Although these are a small proportion of all queries, they are almost always bad to expand.

The results of applying this uniform threshold-setting policy to all of the collections we have studied are shown in Table 4, along with the breakdown on the expansion improvement for all queries above the threshold. We would not expect to see large improvements in mean average precision since this method is aimed at improving the consistency of retrieval by identifying a small number of very bad-to-expand queries. For certain collections (e.g. TREC 6 and TREC 8) the effect of the method, even with this automatic thresholding is quite good: all the queries above the threshold are bad-to-expand or neutral so some important poor expansions are avoided. This leads to a substantial MAP improvement for these collections over consistent relevance retrieval.

Two of the failures of the model comparison method with automatic thresholding (below the line in Table 4) may show the way towards improving the method.

One mode of failure is seen in the case of TREC 7. Here there are only 3 very bad-to-expand queries (Δ average precision less that -0.1) using relevance model retrieval. So the method has little opportunity to shine in this collection, and only 1 of the 3 really bad to expand queries scores badly enough to be detected with the automatic threshold. The problem here is related to the small size of the test query set coupled with the fact that our method with the automatic thresholds in tuned to detect small proportions of queries. If we think of construction of the test query set as a random process, our method is hurt here by random fluctuations in that process. Another way of looking at it is that for this particular test query set and collection, rele-

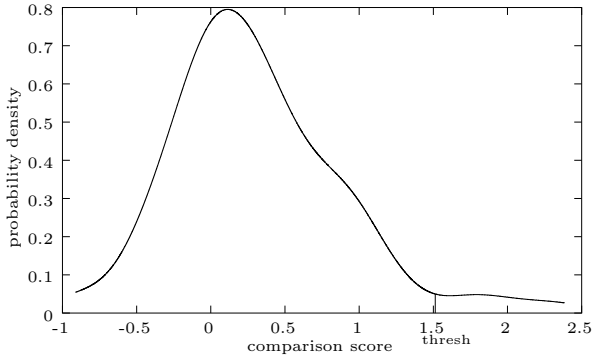

Figure 5: The estimated probability density function for comparison scores in the TREC 8 collection estimated from 100 random one term queries sampled from the vocabulary of TREC 8. The sampled terms are issued as queries and the model comparison is exactly the same way as for user queries. The distribution is estimated using kernel smoothing on the samples. A vertical line labelled "thresh" indicates a model comparison score with 95% of the estimated probability mass below it.

vance model retrieval works particularly well, leaving little room for improvement in the face of any prediction error at all.

Another mode of failure is exhibited in the case of the Query Track aggregate. Here the query sets exhibit high variability. The Query Track aggregate queries range from title-like queries to long natural language queries[4] with lots of different styles.

5. RELATED WORK

Automatic query expansion techniques have been studied extensively (for example [17, 6, 20, 15]). These studies typically mention the degree to which the techniques can lead to very poor performances for some queries as an issue. There also has been much work ([5] for example) on using user feedback to improve retrieval ranked lists. Since these systems can suffer from the same sort of straying from the original sense of the query, they would be candidates for the use of techniques, such as ours, designed to detect such straying. We know of no published works on predicting or sensing automatically when such techniques fail, however.

6. FUTURE WORK

We would like to learn how to tune our model comparisons for optimal effectiveness on any collection and type of query. We would also like to improve the ranked list models and comparison methods. One possibility is to estimate the probability of relevance of a document at a certain rank position in a ranked list from training data.

7. CONCLUSION

We presented a method for improving the consistency of query expansion results. The method provides a framework for measuring when a new ranked list of documents (e.g. from expanded retrieval or feedback) has strayed significantly from the usage of important terms in an original ranked list of documents (e.g. from unexpanded retrieval). In the case of expansion, *not* using the expansion results in these cases will usually avoid one type of expansion failure by sensing that something has gone wrong.

For language modeling based retrieval, we have shown on a variety of test collections that the method can generally improve the consistency (and even the mean average precision) by catching a small proportion of queries and avoiding some poor expansions.

This work suggests one meaningful criterion that systems may use to help avoid showing users the results of techniques that prove bad for a particular query. We believe that systems meaningfully distinguishing between queries provides a very promising approach for improving the performance of real information retrieval systems.

Collection	Rel Model	Model Comp top terms	Above Threshold good	neut	bad
TREC 6	0.2013	0.2197	0	3	3
TREC 8	0.2715	0.2812	0	0	3
AP88-90	0.2875	0.2894	0	4	1
TREC 5	0.1609	0.1621	0	2	1
QT agg	0.2219	0.2217	13	32	11
TREC 1+2+3	0.2490	0.2451	1	0	0
TREC 7	0.2524	0.2394	2	1	1

Table 4: Mean average precision (MAP) for model comparison-based selective expansion in contrast to doing relevance retrieval all the time. The threshold is set at a point where 95% of random one-term queries from the vocabulary would get a lower score. The method improves the MAP for test sets in the top half of the table (above the line) and slightly hurts average precision performance for test sets listed below the line. The breakdown is given for all queries above the threshold, based on the average precision change on expansion, Δ, where "good" means average precision $\Delta > 0.05$, "neut" means $|\Delta| < 0.05$ and "bad" means $\Delta < -0.05$. The queries used for each collection are the same as given in Table 1.

8. ACKNOWLEDGEMENTS

We thank Victor Lavrenko for discussions of best smoothing practices and David Fisher for technical help and advice. We are also indebted to Andrés Corrada-Emmanuel for explaining the analyses done at the NRRC RIA workshop. This work was supported in part by the Center for Intelligent Information Retrieval, in part by NSF grant #IIS-9907018, and in part by SPAWARSYSCEN-SD grant number N66001-02-1-8903. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsors.

9. REFERENCES

[1] C. Arndt. *Information measures : information and its description in science and engineering*. Springer, Berlin, New York, 2001.

[2] A. Bowman and A. Azzilini. *Applied Smoothing Techniques for Data Analysis*. Oxford University Press, New York, 1997.

[3] J. Broglio, J. P. Callan, and W. B. Croft. INQUERY system overview. In *Proc. TIPSTER Text Program (Phase I)*, pages 47–67. Morgan Kaufmann, 1994.

[4] C. Buckley. The trec-9 query track. In E. Voorhees and D. Harman, editors, *Proceedings of the Ninth Text REtrieval Conference(TREC-9)*, 2000. NIST Special Publication 500-249.

[5] C. Buckley and G. Salton. Optimization of relevance feedback weights. In *Proc. of the 18th Annual ACM SIGIR Conference*, pages 351–357, July 1995.

[6] C. Buckley, G. Salton, J. Allan, and A. Singhal. Automatic query expansion using SMART: TREC 3. In *Text REtrieval Conference*, pages 0–, 1994.

[7] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, New York, 1991.

[8] W. B. Croft and J. Lafferty, editors. *Language Modeling for Information Retrieval*, pages 11–56. Kluwer, 2003.

[9] W. B. Croft and J. Lafferty, editors. *Language Modeling for Information Retrieval*. Kluwer Academic, Dordrecht, 2003.

[10] S. Cronen-Townsend and W. B. Croft. Quantifying query ambiguity. In *Proc. of Human Language Technology 2002*, pages 94–98, March 2002.

[11] S. Cronen-Townsend, Y. Zhou, and W. B. Croft. Predicting query performance. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 299–306, August 2002.

[12] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, New York, 1973.

[13] R. Krovetz. Viewing morpholgy as an inference process. In *Proc. of the 16th Annual ACM SIGIR Conference*, pages 191–202, June–July 1993.

[14] V. Lavrenko. Personal communication.

[15] V. Lavrenko and W. B. Croft. Relevance-based language models. In *Research and Development in Information Retrieval*, pages 120–127, 2001.

[16] P. Ogilvie and J. Callan. Experiments using the lemur toolkit. In *Proc. of the Tenth Text Retrieval Conference, (TREC-10)*, pages 103–108, 2002.

[17] S. Robertson. On term selection for query expansion. *Journal of Documentation*, 46:359–364, 1984.

[18] F. Song and W. B. Croft. A general language model for informatioon retrieval. In *Proceedings of the 22nd annual international ACM SIGIR conference*, pages 279–280, 1999.

[19] H. C. Taneja and R. K. Tuteja. Characterization of a quantitative-qualitative measure of relative information. *Information Sciences*, 33:217–222, 1984.

[20] J. Xu and W. B. Croft. Improving the effectiveness of information retrieval with local context analysis. *ACM Transactions on Information Systems*, 18(1):79–112, 2000.

[21] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Research and Development in Information Retrieval*, pages 334–342, 2001.