A MAXIMUM LIKELIHOOD RATIO INFORMATION RETRIEVAL MODEL

Kenney Ng

Spoken Language Systems Group
MIT Laboratory for Computer Science
545 Technology Square, Cambridge, MA 02139 USA

ABSTRACT

In this paper we present a novel probabilistic information retrieval model that scores documents based on the relative change in the document likelihoods, expressed as the ratio of the conditional probability of the document given the query and the prior probability of the document before the query is specified. The document likelihoods are computed using statistical language modeling techniques and the model parameters are estimated automatically and dynamically for each query to optimize well-specified (maximum likelihood) objective functions. We derive the basic retrieval model, describe the details of the model, and present some extensions to the model including a method to perform automatic feedback. Development experiments are performed using the TREC-6 ad hoc text retrieval task and performance is measured using the TREC-7 ad hoc task. Official evaluation results on the 1999 TREC-8 ad hoc task are also reported. The performance results demonstrate that the model is competitive with current state-of-the-art retrieval approaches.

1. INTRODUCTION

Probabilistic modeling for information retrieval (IR) has a long history [3]. Many of these approaches try to evaluate the probability of a document being relevant (R) to a given query Q by estimating $\mathrm{p}(R|Q, D_i)$ for every document D_i in the collection. These relevance probabilities are then used to rank order the retrieved documents. However, due to the imprecise definition of the concept of relevance and the lack of available relevance training data, reliably estimating these probabilities has been a difficult task. Because of the the nature of the IR task, training data in the form of document-query pairs labeled with their corresponding relevance judgments is not generally available *a priori*. Previously seen queries, for which relevance information can be created, can be used for training but their applicability to new queries is not clear. Some relevance information can be obtained in a multi-pass retrieval strategy by using relevance feedback. However, only a small number of relevance judgments is typically generated. Many of these probabilistic methods are better suited for related applications, such as information filtering, where more relevance training data is available [6, 7].

Instead of the imprecisely defined notion of relevance, we consider the better defined measure of likelihood. In particular, we examine the relative change in the likelihood of a document before and after a query is specified, and use that as the metric for scoring and ranking the documents. The idea is that documents that become more likely after the query is specified are probably more useful to the user and should score better and be ranked ahead of those documents whose likelihoods either stay the same or decrease. The document likelihoods are computed using statistical language modeling techniques and the model parameters are estimated automatically and dynamically for each query to optimize well-specified objective functions.

The paper is organized as follows. In Section 2, we derive the basic retrieval model, describe the details of the model, and present some extensions to the model including a method to perform automatic feedback. We also discuss some related modeling approaches. Next, in Section 3, we evaluate the performance of the retrieval model and present experimental results on the TREC-6 ad hoc text retrieval task. Then, in Section 4, we objectively evaluate the system on the TREC-7 ad hoc task and report official evaluation results on the 1999 TREC-8 ad hoc task. Finally, we close with a summary in Section 5.

2. INFORMATION RETRIEVAL MODEL

Given a collection of n documents, $\{D_i\}_{i=1}^n$, each document D_i has a prior likelihood given by $\mathrm{p}(D_i)$. After a query Q is specified by a user, the likelihood of each document changes and becomes that given by the conditional probability: $\mathrm{p}(D_i|Q)$. Some documents will become more likely after the query is specified while others will either remain the same or become less likely. The documents that become more likely are probably more useful to the user and should score better and be ranked ahead of those that either stay the same or become less likely. As a result, we propose to use the relative change in the document likelihoods, expressed as the likelihood ratio of the conditional and the prior probabilities, as the metric for scoring and ranking the documents in response to query Q:

$$\mathrm{S}(D_i, Q) = \frac{\mathrm{p}(D_i|Q)}{\mathrm{p}(D_i)} \tag{1}$$

We can decompose this likelihood ratio score into more easily estimated components using Bayes' Rule and rewrite (1) as:

$$\mathrm{S}(D_i, Q) = \frac{\mathrm{p}(Q|D_i)\,\mathrm{p}(D_i)/\mathrm{p}(Q)}{\mathrm{p}(D_i)} = \frac{\mathrm{p}(Q|D_i)}{\mathrm{p}(Q)} \tag{2}$$

where $\mathrm{p}(Q|D_i)$ is the probability of query Q given document D_i and $\mathrm{p}(Q)$ is the prior probability of query Q. Each document D_i specifies a different language model Λ_i. We can view $\mathrm{p}(Q|D_i)$ as the probability that query Q is generated by Λ_i, the language model associated with document D_i. This means that our goal during the retrieval process is to find those documents in the collection that maximize the likelihood of the query. These documents should be the ones that are most useful to the user who specified query Q.

The $\mathrm{p}(Q)$ term represents the probability that query Q is generated from a document independent (general) language model Λ, and serves as a normalization factor. Since $\mathrm{p}(Q)$ is constant for all documents D_i given a specific query Q, it does not affect the ranking of the documents and can be safely removed from the scoring function. However, this $\mathrm{p}(Q)$ normalization factor is useful if we want a meaningful interpretation of the scores (as a relative change in the likelihood) and if we want to be able to compare scores across different queries. In Section 3.3, we illustrate the usefulness of $\mathrm{p}(Q)$ for these purposes. In addition, the $\mathrm{p}(Q)$ normalization factor is an important part of the automatic feedback extension to the basic model as we will see in Section 2.2. For these reasons, we will keep the $\mathrm{p}(Q)$ term in the scoring function in (2).

2.1. Model Details

In order to compute the score in (2), we need to be able to estimate the quantities $\mathrm{p}(Q|D_i)$ and $\mathrm{p}(Q)$. To do this, we make the assumption that the query Q is drawn from a multinomial distribution over the set of possible terms in the corpus and document D_i specifies the parameters of the multinomial model. This gives us the following estimates for $\mathrm{p}(Q|D_i)$ and $\mathrm{p}(Q)$:

$$\mathrm{p}(Q|D_i) = \frac{n!}{\prod_{t=1}^{k} c_t!} \prod_{t=1}^{k} \mathrm{p}(t|D_i)^{c_t} \quad (3)$$

$$\mathrm{p}(Q) = \frac{n!}{\prod_{t=1}^{k} c_t!} \prod_{t=1}^{k} \mathrm{p}(t)^{c_t} \quad (4)$$

where c_t is the number of times term t occurs in query Q, k is the number of distinct terms in the corpus, $n = \sum_{t=1}^{k} c_t$ is the total number of terms in query Q, $\mathrm{p}(t|D_i)$ is the probability of query term t occurring in document D_i with the constraint $\sum_{t=1}^{k} \mathrm{p}(t|D_i) = 1$, and $\mathrm{p}(t)$ is the probability of query term t occurring in the document collection with the constraint $\sum_{t=1}^{k} \mathrm{p}(t) = 1$. Substituting (3) and (4) into (2) and simplifying (noting that $c_t! = 1$ for $c_t = 0$), we have:

$$\mathrm{S}(D_i, Q) = \prod_{t=1}^{k} \left(\frac{\mathrm{p}(t|D_i)}{\mathrm{p}(t)} \right)^{c_t} \quad (5)$$

Since $x^0 = 1$ for all x, the product over all k terms can be replaced by a product over only the terms that occur in the query:

$$\mathrm{S}(D_i, Q) = \prod_{t \in Q} \left(\frac{\mathrm{p}(t|D_i)}{\mathrm{p}(t)} \right)^{c_t} \quad (6)$$

To simplify computation and to prevent numerical underflows, we perform the score computation in the log domain:

$$\mathrm{S}_l(D_i, Q) = \log\, \mathrm{S}(D_i, Q) = \sum_{t \in Q} c_t \, \log \left(\frac{\mathrm{p}(t|D_i)}{\mathrm{p}(t)} \right) \quad (7)$$

We note that since the logarithm is a monotonic transformation, the rank ordering of the documents using the log score remains the same as that using the original score.

In the original multinomial model, c_t is the number of times term t occurs in query Q and can only take on integral values: $c_t = 0, 1, \ldots, n$. We would like to generalize c_t so that it can take on non-negative real values. This will allow more flexible weighting of the query terms including the use of fractional counts which will be useful in our automatic relevance feedback extension (Section 2.2) and query section weighting (Section 3.6). To indicate this generalization in the scoring function, we replace c_t in (7) with $q(t)$, which can be interpreted as the weight of term t in query Q:

$$\mathrm{S}_l(D_i, Q) = \sum_{t \in Q} q(t) \, \log \left(\frac{\mathrm{p}(t|D_i)}{\mathrm{p}(t)} \right) \quad (8)$$

This generalization does not affect the ranking of the documents since it is equivalent to adding a query-dependent constant multiplicative factor, $1/n$, to the score in (7) to convert the c_t counts to the $q(t)$ numbers. In fact, we can interpret $q(t)$ as $\mathrm{p}(t|Q)$, the probability of term t occurring in query Q, if $q(t) = c_t/n$ where $n = \sum_t c_t$.

We note that the scoring function in (8) can be related to the Kullback-Leibler distance [2], which is an information theoretic measure of the divergence of two probability distributions $\mathrm{p}_1(x)$ and $\mathrm{p}_2(x)$:

$$KL(\mathrm{p}_1(x), \mathrm{p}_2(x)) = -\sum_{x} \mathrm{p}_2(x) \log \left(\frac{\mathrm{p}_1(x)}{\mathrm{p}_2(x)} \right) \quad (9)$$

To show this relationship, we start by rewriting (8) as follows:

$$\mathrm{S}_l(D_i, Q) = \sum_{t \in Q} q(t) \, \log \mathrm{p}(t|D_i) - \sum_{t \in Q} q(t) \, \log \mathrm{p}(t) \quad (10)$$

Next, we add in and subtract out $\sum_{t \in Q} q(t) \, \log q(t)$, rearrange terms, and then collapse terms to get:

$$\mathrm{S}_l(D_i, Q) = \underbrace{\sum_{t \in Q} q(t) \, \log \left(\frac{\mathrm{p}(t|D_i)}{q(t)} \right)}_{-KL(q(t), \mathrm{p}(t|D_i))} - \underbrace{\sum_{t \in Q} q(t) \, \log \left(\frac{\mathrm{p}(t)}{q(t)} \right)}_{+KL(q(t), \mathrm{p}(t))} \quad (11)$$

Recall that $q(t)$ can be interpreted as $\mathrm{p}(t|Q)$, the probability of term t in query Q, $\mathrm{p}(t|D_i)$ is the probability of term t in document D_i, and $\mathrm{p}(t)$ is the probability of term t in the general language (i.e., using a document-independent model). The first term in (11) is the (negative) KL divergence between the term distribution of query Q and document D_i. If the two term distributions are identical, then the divergence will be zero. As the difference between the query and document distributions becomes greater, the divergence increases, and the score decreases (because of the negative sign on the term). The second term is the KL divergence between the term distribution of query Q and a general document-independent model. Since this term doesn't depend on the document, it has no effect on the rankings of the retrieved documents; it only serves as a bias or normalization factor. It is query-dependent and only comes into play if we compare scores across different queries.

We also note that the scoring function in (8) has the form of the standard vector space model. It consists of the sum over all terms t in the query of the product of a query dependent factor, $q(t)$, and a document dependent factor, $\log\left(\mathrm{p}(t|D_i)/\mathrm{p}(t)\right)$. It turns out that many probabilistic models can be expressed in the standard vector space model format [3, 9, 15]. The models differ in what the query and document factors are and how they are estimated.

Next, we need to estimate the probabilities $\mathrm{p}(t|D_i)$ and $\mathrm{p}(t)$. We start by considering their maximum likelihood (ML) estimates:

$$\mathrm{p}_{\mathrm{ml}}(t|D_i) = \frac{d_i(t)}{\sum_{t=1}^{k} d_i(t)} \quad (12)$$

$$\mathrm{p}_{\mathrm{ml}}(t) = \frac{\sum_{i=1}^{n} d_i(t)}{\sum_{i=1}^{n} \sum_{t=1}^{k} d_i(t)} \quad (13)$$

where $d_i(t)$ is the number of occurrences of term t in document D_i, k is the number of distinct terms in the corpus, and n is the number of documents in the collection.

With a large document collection, there is enough data for $\mathrm{p}_{\mathrm{ml}}(t)$ to be robustly estimated. However, this ML estimate will assign a probability of zero to terms that do not occur in the document collection. To avoid this undesirable property, we can use Good-Turing (GT) methods to estimate $\mathrm{p}(t)$ [10]. GT methods provide probability estimates for both observed and unobserved terms with the constraint that the total probability of all terms must sum to one. For unobserved terms, GT methods provide an estimate of the *total* probability of these terms. This total probability can then be divided among the possible unobserved terms to provide per term probability estimates. For observed terms, GT methods provide probability estimates for these terms that are consistent with estimating non-zero probabilities for the unobserved terms. This is done by reducing the total probability of the observed terms to be less than one. Good-Turing methods work as follows. If a certain term t occurs r times in the document collection, the ML estimate of $\mathrm{p}(t)$ is given by:

$$\mathrm{p}_{\mathrm{ml}}(t) = r/N \quad (14)$$

where N is the total number of terms observed in the document collection. With GT estimation, the count r is replaced by a modified count r^* which is calculated as:

$$r^* = (r+1)\frac{N_{r+1}}{N_r} \quad (15)$$

where N_r is the number of terms that occurs exactly r times in the document collection. As a result, the GT estimate of $\mathrm{p}(t)$ for observed terms is given by:

$$\mathrm{p}_{\mathrm{gt}}(t) = \mathrm{p}_r = r^*/N \tag{16}$$

where $N = \sum_r rN_r$ is the total number of terms observed in the document collection. The GT estimate for the *total* probability of unobserved terms is given by:

$$\mathrm{p}_0 = N_1/N \tag{17}$$

This total probability is then divided equally among the possible unobserved terms to provide per term probability estimates. Using the observed N_r values to calculate r^* in (15) can become problematic if $N_r = 0$ for some r. As a result, it is necessary to pre-smooth N_r so that it never equals zero. There are many different possible smoothing methods and each gives rise to a slightly different GT approach. We use the Simple Good-Turing (SGT) approach described in [5]. Basically N_r is linearly smoothed (in the log domain) and a decision rule is used to decide when to switch from using the observed N_r values to the smoothed values.

Unlike the estimate for $\mathrm{p}(t)$, the quantity $\mathrm{p}_{\mathrm{ml}}(t|D_i)$ is likely to be poorly estimated regardless of the size of the document collection because of the limited size of the individual documents. Many of the terms in the model will have zero probability. There are many different ways to compensate for this sparse data problem. One approach is to model the term distributions using parametric distributions such as Beta and Dirichlet distributions. A standard statistical language modeling approach, and the one we adopt, is to linearly interpolate the more detailed $\mathrm{p}_{\mathrm{ml}}(t|D_i)$ model with a better estimated, but more general model, for example, $\mathrm{p}_{\mathrm{gt}}(t)$ [10]:

$$\mathrm{p}(t|D_i) = \alpha\, \mathrm{p}_{\mathrm{ml}}(t|D_i) + (1-\alpha)\, \mathrm{p}_{\mathrm{gt}}(t) \tag{18}$$

where α is the mixture weight. The estimate-maximize (EM) algorithm [4] can be used to estimate α to maximize the (log) likelihood of query Q given document D_i:

$$\begin{aligned}\alpha^* &= \arg\max_{\alpha} \log\left(\mathrm{p}(Q|D_i)\right) &(19)\\ &= \arg\max_{\alpha} \sum_{t\in Q} q(t)\, \log\left(\alpha\, \mathrm{p}_{\mathrm{ml}}(t|D_i) + (1-\alpha)\, \mathrm{p}_{\mathrm{gt}}(t)\right) &(20)\end{aligned}$$

In the above formulation, there is a different α for each document D_i. To simplify the model and to provide more data for parameter estimation, we can "tie" the α weight across the documents so that there is only a single, document-independent, α for each query Q. The following iterative procedure can then be used to estimate α:

1. Initialize α to a random estimate between 0 and 1.
2. Update α using:

$$\begin{aligned}\alpha' \;=\; &\frac{1}{\sum_{t\in Q}\sum_{i\in\mathcal{I}_Q} q(t)} \times \\ &\sum_{t\in Q}\sum_{i\in\mathcal{I}_Q} q(t)\, \frac{\alpha\, \mathrm{p}_{\mathrm{ml}}(t|D_i)}{\alpha\, \mathrm{p}_{\mathrm{ml}}(t|D_i) + (1-\alpha)\, \mathrm{p}_{\mathrm{gt}}(t)}\end{aligned}$$

3. If α has converged (i.e., $|\alpha' - \alpha| < \delta$ for some small threshold δ) then stop. Otherwise, set $\alpha = \alpha'$ and goto step 2.

In this procedure, $\mathcal{I}_Q$ contains the indices of the set of documents used to estimate α for query Q. We need to decide which documents should be in this set. If we use *all* the documents in the collection (i.e., $\mathcal{I}_Q = \{1,\ldots,n\}$), the query terms will occur so seldomly in the entire collection that α will almost always be set to zero. That would not be very useful. What we want is a reasonable estimate of α for those documents that are likely to be relevant to the query since they are the ones that we are interested in. Ideally, we want the set of documents to be those that *are* relevant to query Q. However, since this information is not available, we need to use an approximation. One approach is to borrow the technique used in automatic relevance feedback [15] (see Section 2.2). Basically, we perform a preliminary retrieval run using an initial guess for α (e.g., $\alpha = 0.5$) and assume that the top M retrieved documents are relevant to the query. These M top-scoring documents then become the set we use to estimate the α weight for query Q. $M = 5$ is a typical value that we use.

Using the approach described above, a separate α is estimated for each query Q. If desired, one can pool the query terms across all the queries and estimate a single query-independent α. It is important to note that the above procedure estimates the mixture parameters dynamically using the current query and the current document collection. This is in contrast to the standard approach of determining static, query-independent, model parameter values by empirically tuning on an old development set which typically consists of a different set of queries and potentially a different collection of documents. In Section 3.4, we explore the effect of different estimated α values on retrieval performance and examine query-specific and query-independent α's.

In summary, the final metric used for scoring document D_i in response to query Q is obtained by substituting the estimates for $\mathrm{p}(t)$ and $\mathrm{p}(t|D_i)$ (Equations 16 and 18, respectively) into (8):

$$\mathrm{S}_l(D_i, Q) = \sum_{t\in Q} q(t)\, \log\left(\frac{\alpha\, \mathrm{p}_{\mathrm{ml}}(t|D_i) + (1-\alpha)\, \mathrm{p}_{\mathrm{gt}}(t)}{\mathrm{p}_{\mathrm{gt}}(t)}\right) \tag{21}$$

2.2. Automatic Relevance Feedback

Automatic relevance feedback is a proven method for improving information retrieval performance [6]. The process works in three steps. First, the original query is used to perform a preliminary retrieval run. Second, information from these retrieved documents are used to automatically construct a new query. Third, the new query is used to perform a second retrieval run to generate the final results. A commonly used query reformulation strategy, the Rocchio algorithm [15], starts with the original query, Q, then adds terms found in the top N_t retrieved documents and subtracts terms found in the bottom N_b retrieved documents to come up with a new query, Q'. Modifying the query in this way adds new terms that occur in documents that are likely to be relevant to the query and eliminates terms that occur in documents that are probably non-relevant. The query terms are also reweighted. The goal is to improve the ability of the query to discriminate between relevant and non-relevant documents.

We extend our basic retrieval model to include an automatic relevance feedback processing stage by developing a new query reformulation algorithm that is specific to our probabilistic model. Recall that in our retrieval model, we score document D_i in response to query Q using the likelihood ratio score (2):

$$\mathrm{S}(D_i, Q) = \frac{\mathrm{p}(Q|D_i)}{\mathrm{p}(Q)} \tag{22}$$

Since the documents are ranked based on descending values of this score, we can view the goal of the automatic feedback procedure as trying to create a new query Q' (based on the original query Q and the documents retrieved from the preliminary retrieval pass) such that the score using the new query is better than the score using the original query for those documents D_i that are relevant to the query:

$$\frac{\mathrm{p}(Q'|D_i)}{\mathrm{p}(Q')} \geq \frac{\mathrm{p}(Q|D_i)}{\mathrm{p}(Q)} \qquad \text{for } i \in \mathcal{I}_Q \tag{23}$$

Because $\mathcal{I}_Q$, the set of relevant documents for query Q, is not known, we use an approximation and assume that the top scoring documents from a preliminary retrieval run using the original query are relevant.

There are many different ways to decide which of the top scoring documents to select. One approach is to simply select a fixed number, M, of the top scoring documents. One concern with this approach is that the selected documents can have very disparate scores. There can be a big score difference between the first and the M^{th} document. Another approach is to use an absolute score threshold, θ, so only documents with scores above θ are selected. With this approach, it is possible to not have any documents that score above the threshold. A different approach, and the one we adopt, is to use a relative score threshold, $\gamma \leq 1$, so documents that score within a factor of γ of the top scoring document are selected:

$$\text{select } D_i \text{ if } \qquad \frac{\mathrm{S}(D_i, Q)}{\max\limits_{D_i} \mathrm{S}(D_i, Q)} \leq \gamma \tag{24}$$

This approach results in a variable number of documents for each query, but the selected documents will have similar scores. A typical threshold value that we use is $\gamma = 0.75$.

Since we want to improve the score for all the documents in the set $\mathcal{I}_Q$ simultaneously, we need to deal with the set of documents jointly. One way to do this is to create a new joint document D' by pooling together all the documents in the set $\mathcal{I}_Q$ so the number of occurrences of term t in the joint document D' is given by:

$$d'(t) = \sum_{i \in \mathcal{I}_Q} d_i(t) \tag{25}$$

Another variation is to weight the contribution of each document, D_i, by its preliminary retrieval score, $\mathrm{S}(D_i, Q)$, so documents that score better have more of an impact:

$$d'(t) = \sum_{i \in \mathcal{I}_Q} \mathrm{S}(D_i, Q)\, d_i(t) \tag{26}$$

Using this new joint document, D', the inequality in (23) becomes:

$$\frac{\mathrm{p}(Q'|D')}{\mathrm{p}(Q')} \geq \frac{\mathrm{p}(Q|D')}{\mathrm{p}(Q)} \tag{27}$$

Substituting our models for the conditional and prior probabilities and working in the log domain (Equation 8), we have

$$\sum_{t \in Q'} q'(t) \, \log \left(\frac{\mathrm{p}(t|D')}{\mathrm{p}(t)} \right) \geq \sum_{t \in Q} q(t) \, \log \left(\frac{\mathrm{p}(t|D')}{\mathrm{p}(t)} \right) \tag{28}$$

Let us consider the creation of the new query Q' in two steps. First, let us examine which terms should be *removed* from the original query Q in order to improve the score. Second, we can then examine which terms from the joint document D' should be *added* to the query to further improve the score.

Starting with the original query Q, we consider each query term t and determine whether it should be included or excluded from the new query Q'. Since the query term weights $q(t)$ are constrained to be greater than zero, the only way that a query term t can decrease the score is if $\frac{\mathrm{p}(t|D')}{\mathrm{p}(t)} < 1$. Therefore, if we exclude such terms from the new query Q' (while keeping the term weights the same, i.e., $q'(t) = q(t)$), we can be assured that the inequality in (28) is satisfied. This selection criteria makes intuitive sense since it basically states that query terms that occur more frequently in the general collection than in the pooled document D' (which is created from assumed relevant documents) should not be used.

Next, we consider which terms from the joint document D' should be included to the query Q' in order to further improve the score. Following the same arguments as those used above, and noting that $q'(t) > 0$, we see that only terms t for which $\frac{\mathrm{p}(t|D')}{\mathrm{p}(t)} > 1$ can increase the score. As a result, we will only add those terms from D' that satisfy this property. Using this term selection criteria, we maintain the inequality in (28) with each newly included term. Substituting the estimates for $\mathrm{p}(t)$ and $\mathrm{p}(t|D_i)$ (Equations 16 and 18, respectively), the term selection criteria becomes:

$$\frac{\mathrm{p}(t|D')}{\mathrm{p}(t)} > 1 \tag{29}$$

$$\frac{\alpha\, \mathrm{p}_{\mathrm{ml}}(t|D') + (1-\alpha)\, \mathrm{p}_{\mathrm{gt}}(t)}{\mathrm{p}_{\mathrm{gt}}(t)} > 1$$

$$\frac{\mathrm{p}_{\mathrm{ml}}(t|D')}{\mathrm{p}_{\mathrm{gt}}(t)} > 1 \tag{30}$$

Therefore, we can equivalently use $\frac{\mathrm{p}_{\mathrm{ml}}(t|D')}{\mathrm{p}_{\mathrm{gt}}(t)} > 1$ or $\log\left(\frac{\mathrm{p}_{\mathrm{ml}}(t|D')}{\mathrm{p}_{\mathrm{gt}}(t)}\right) > 0$ to perform the term selection.

The only issue that remains is the estimation of appropriate values for the weights $q'(t)$ of the newly included query terms. Since the value of the score can be increased arbitrarily by using increasingly larger values of $q'(t)$, we need to constrain the aggregate value of the weights. One reasonable constraint is that the magnitude of the query weights be unity:

$$||Q'|| = \sqrt{\sum_{t \in Q'} q'(t)^2} = 1 \tag{31}$$

Adopting this constraint, we can use the technique of Lagrange multipliers [1] to find the set of query term weights, $\{q'(t)\}$, that maximizes the score:

$$\sum_{t \in Q'} q'(t) \, \log \left(\frac{\mathrm{p}(t|D')}{\mathrm{p}(t)} \right) \tag{32}$$

The corresponding Lagrangian function is given by:

$$Ł(Q', \lambda) = \sum_{t \in Q'} q'(t) \, \log \left(\frac{\mathrm{p}(t|D')}{\mathrm{p}(t)} \right) + \lambda \left(\sqrt{\sum_{t \in Q'} q'(t)^2} - 1 \right) \tag{33}$$

Taking the partial derivative of (33) with respect to λ and setting it to zero, we get back the constraint equation:

$$\frac{\partial}{\partial \lambda} Ł(Q', \lambda) = 0 \tag{34}$$

$$\sqrt{\sum_{t \in Q'} q'(t)^2} = 1 \tag{35}$$

Taking the partial derivative of (33) with respect to the query term weight $q'(t)$ and setting it to zero, we get

$$\frac{\partial}{\partial q'(t)} Ł(Q', \lambda) = 0 \tag{36}$$

$$\log \left(\frac{\mathrm{p}(t|D')}{\mathrm{p}(t)} \right) + \lambda \frac{q'(t)}{\sqrt{\sum_{t \in Q'} q'(t)^2}} = 0 \tag{37}$$

Taking the second derivative, we get

$$\frac{\partial^2}{\partial q'(t)^2} Ł(Q', \lambda) = \lambda \left(1 - q'(t)^2 \right) \tag{38}$$

For the score to be maximized, we need this second derivative to be less than zero. Since $0 < q'(t) < 1$, we must have $\lambda < 0$ in order for (38) to be negative.

Combining equations (35) and (37) and solving for $q'(t)$, we get

$$q'(t) = -\frac{1}{\lambda}\log\left(\frac{\mathrm{p}(t|D')}{\mathrm{p}(t)}\right) \tag{39}$$

Since we require $\lambda < 0$, we see that the appropriate query weights simply have to be proportional to their score contribution:

$$q'(t) \propto \log\left(\frac{\mathrm{p}(t|D')}{\mathrm{p}(t)}\right) \tag{40}$$

This weighting scheme makes intuitive sense since we want to emphasize terms that contribute more to the score. If desired, we can determine the exact value of the proportionality factor by substituting (39) back into (35) and solving for λ. Doing this, we find that:

$$\lambda = -\sqrt{\sum_{t \in Q'}\left(\log\left(\frac{\mathrm{p}(t|D')}{\mathrm{p}(t)}\right)\right)^2} \tag{41}$$

Our description of the automatic relevance feedback procedure is now complete. We have a procedure that automatically creates a new query Q' based on the original query Q and a set of top-ranked documents retrieved from a preliminary retrieval pass. The goal of the procedure is to increase the likelihood ratio scores of the top-ranked documents by removing certain terms from the original query and adding new terms from the top-ranked documents with appropriate term weights. Hopefully improving the scores will lead to improved information retrieval performance.

We note that this automatic feedback procedure significantly increases the number of terms in the query since many of the terms in the joint document D' will satisfy the selection criteria (29). If desired, one can limit the number of additional terms by modifying this term selection criteria so only terms with scores greater than some threshold $\phi \geq 1$ will be included:

$$\text{add term } t \text{ if} \qquad \frac{\mathrm{p}(t|D')}{\mathrm{p}(t)} > \phi \tag{42}$$

In Section 3.5, we examine the ability of this automatic relevance feedback procedure to improve retrieval performance and explore the effects of limiting the number of new query terms by increasing the value of ϕ in (42).

2.3. Related Work

In our retrieval model, we use the relative change in the likelihood of a document D_i before and after the user query Q is specified, expressed as the likelihood ratio of the conditional and the prior probabilities, $\frac{\mathrm{p}(D_i|Q)}{\mathrm{p}(D_i)}$, as the metric for scoring and ranking the documents. A document that becomes more likely after the query is specified is probably more useful to the user than one that either remains the same or becomes less likely. This score can be equivalently rewritten as $\frac{\mathrm{p}(Q|D_i)}{\mathrm{p}(Q)}$. Since we need to estimate $\mathrm{p}(Q|D_i)$, the probability of query Q given document D_i, our model is related to several recently proposed IR approaches which also make use of this probabilistic quantity [9, 11, 12].

In [12] and [9], a language modeling argument is used to directly posit that $\mathrm{p}(Q|D_i)$ is an appropriate quantity for scoring document D_i in response to query Q. Mixture models are then used to compute this quantity. In [11], the probability that document D_i is relevant given query Q, $\mathrm{p}(D_i \,\mathrm{is}\, R|Q)$, is used to score the documents. This quantity can be rewritten, using Bayes Rule, as $\frac{\mathrm{p}(Q|D_i \,\mathrm{is}\, R)\,\mathrm{p}(D_i \,\mathrm{is}\, R)}{\mathrm{p}(Q)}$. A generative hidden Markov model (HMM) is then used to compute the quantity $\mathrm{p}(Q|D_i \,\mathrm{is}\, R)$.

Although our retrieval model shares this commonality with these other approaches, there are some important differences. First, as described above, our model is derived starting from a different theoretical justification. Second, different modeling assumptions and estimation techniques are used to determine the underlying probabilistic quantities. Although we use the standard technique of mixture models to estimate the quantity $\mathrm{p}(Q|D_i)$, the underlying probabilistic components in our mixture model are different from those used in [12] and [9]. We back-off to the term's probability of occurrence in the entire document collection. In [9], the back-off is to the term's document frequency while in [12] the back-off is a scaled version of the term's mean probability of occurrence in documents that contain the term. We also automatically estimate the mixture model parameters dynamically (for each query Q) to maximize the likelihood of the query given a set of top scoring documents $\{D_i\}$ from the current document collection. This is in contrast to the standard approach of determining static, query-independent, mixture model parameter values by empirically tuning on an old development set. In addition, we attempt to deal with unobserved query terms in a more principled way by using Good-Turing techniques to smooth the underlying probability models. Finally, we develop a new automatic relevance feedback strategy that is specific to our probabilistic model. The procedure automatically creates a new query (based on the original query and a set of top-ranked documents from a preliminary retrieval pass) that optimizes a well-specified objective function. In particular, the term selection and the term weight estimation procedures are designed to maximize the likelihood ratio scores of the set of documents presumed to be relevant to the query. Hopefully, improving these scores will lead to improved retrieval performance.

3. INFORMATION RETRIEVAL EXPERIMENTS

Our information retrieval model is evaluated on the TREC-6, TREC-7, and TREC-8 ad hoc text retrieval tasks [6–8]. The ad hoc task involves searching a static set of documents using new queries and returning an ordered list of documents ranked according to their relevance to the query. The retrieved documents are then evaluated against relevance assessments created for each query.

Retrieval performance is measured in terms of a tradeoff between *precision* and *recall*. Precision is the number of relevant documents retrieved over the total number of documents retrieved. Recall is the number of relevant documents retrieved over the total number of relevant documents in the collection. Because it may be cumbersome to compare the performance of different systems using precision-recall curves, a single number performance measure called *mean average precision* (mAP) is commonly used [6]. It is computed by averaging the precision values at the recall points of all relevant documents for each query and then averaging those across all the test set queries.

In this section, we briefly describe the data corpus that comprise the TREC-6, TREC-7, and TREC-8 tasks, mention the text preprocessing that was done, and then present several retrieval experiments using the TREC-6 task. In these development experiments, we explore the usefulness of the $\mathrm{p}(Q)$ normalization in the scoring, the effect of using different mixture weights in the probability model, the use of the automatic relevance feedback processing, and section-based weighting of the query terms.

3.1. Data Corpus

The document collection in the TREC-6, TREC-7, and TREC-8 ad hoc retrieval tasks consists of text stories from various news and information sources. Details of the composition and size of the collections are given in Table 1. The documents in the TREC-7 task are a subset of those in the TREC-6 task (documents from the *Congressional Record* are excluded from the TREC-7 collection). The document collection used in the TREC-8 task is identical to that used in TREC-7. Each collection contains approximately 2 gigabytes of text from over half a million documents.

Data Set	**Size (MB)**	**# docs**	**Avg. # wrds/doc**
Financial Times (FT)	564	210,158	412.7
Federal Register (FR)	395	55,630	644.7
Congressional Record (CR)	235	27,922	1373.5
FBIS (FBIS)	470	130,471	543.6
L.A. Times (LA)	475	131,896	526.5
TREC-6 (all sources)	2139	556,077	541.9
TREC-7 (4 sources: no CR)	1904	528,155	497.9
TREC-8 (same as TREC-7)	1904	528,155	497.9

Table 1: Statistics for the document collections used in the TREC-6, TREC-7, and TREC-8 ad hoc retrieval tasks.

	# of Words		
Data Set (topic #'s)	**Min**	**Max**	**Avg.**
TREC-6 (301-350)	47	156	88.4
title	1	5	2.7
description	5	62	20.4
narrative	17	142	65.3
TREC-7 (351-400)	31	114	57.6
title	1	3	2.5
description	5	34	14.3
narrative	14	92	40.8
TREC-8 (401-450)	23	98	51.3
title	1	4	2.4
description	5	32	13.8
narrative	14	75	35.1

Table 2: Statistics for the test topics used in the TREC-6, TREC-7, and TREC-8 ad hoc retrieval tasks. There are 50 topics in each retrieval task.

There are 50 queries (also called "topics") for each of the TREC-6, TREC-7, and TREC-8 ad hoc retrieval tasks. Topic numbers 301-350 are used in the TREC-6 task, while 351-400 are used in the TREC-7 task, and 401-450 are used in the TREC-8 task. Each topic consists of three sections: a title, a description, and a narrative. Statistics regarding the size of the topics are shown in Table 2.

In order to evaluate the performance of a retrieval system, relevance assessments must be provided for each topic. In other words, for each topic in the test set, the set of the known relevant documents in the collection needs to be determined. Since there are too many documents for complete manual inspection, an approximate method, known as the "pooling method," is used to find the set of relevant documents [7]. For each topic, a pool of possible relevant documents is first created by taking the top 100 documents retrieved from the various participating systems. Next, each document in this pool is manually assessed to determine its relevance. Finally, those documents that are judged relevant become the "answers" for the topic and are used to conduct the performance evaluations. Summary statistics for the number of relevant documents for the topics in the TREC-6, TREC-7, and TREC-8 ad hoc tasks are shown in Table 3. We note that there is great variability. Some topics have many relevant documents while others have only a few.

	# of Relevant Docs			
Data Set (topic #'s)	**Min**	**Max**	**Avg.**	**Total**
TREC-6 (301-350)	3	474	92.2	4611
TREC-7 (351-400)	7	361	93.5	4674
TREC-8 (401-450)	6	347	94.6	4728

Table 3: Statistics for the number of relevant documents for the topics in the TREC-6, TREC-7, and TREC-8 ad hoc retrieval tasks. There are 50 topics in each retrieval task.

In our retrieval experiments, we use the TREC-6 task as the "development" data set for tuning and optimizing our retrieval model. Most of the contrasting experiments will be done on the TREC-6 task. We reserve the TREC-7 task for use as the "test" data to objectively test our final retrieval model. An official TREC "evaluation" run was done using the TREC-8 task. Following standard practices, we use the entire topic statement (consisting of the title, description, and narrative components) in our retrieval experiments, unless otherwise noted.

3.2. Text Preprocessing

Before a document is indexed, it undergoes a relatively standard set of text preprocessing steps. First, the text is normalized to remove non-alphanumeric characters like punctuation and to collapse case. Next, sequences of individual characters are automatically grouped to create single terms in an "automatic acronym aggregation" stage. For example, the text string "`U. S. A.`" would be converted to "`u s a`" after normalization and then to "`usa`" after acronym aggregation. Stop words, derived from a list of 600 words, are then removed from the document. In addition to standard English function words, certain words frequently used in past TREC topics such as "document," "relevant," and "irrelevant" are also included in the list. Finally, the remaining words are conflated to collapse word variants using an implementation of Porter's stemming algorithm [13]. To maintain consistency, each topic description also undergoes the exact same text preprocessing steps before it is indexed and used to retrieve documents from the collection.

3.3. $\mathrm{p}(Q)$ Normalization

As discussed in Section 2.1, the $\mathrm{p}(Q)$ normalization factor in the scoring function (2) does not affect the ranking of the documents because it is constant for all documents D_i given a specific topic Q. However, we choose to keep this factor because it helps to provide a meaningful interpretation of the scores as a relative change in the likelihood and allows the document scores to be more comparable across different topics. In addition, as we've seen in Section 2.2, the $\mathrm{p}(Q)$ normalization factor plays an important role in the term selection and weighting stages of the automatic relevance feedback procedure.

To illustrate the difference between the (unnormalized) likelihood score ($\mathrm{p}(Q|D_i)$) and the (normalized) likelihood ratio score ($\frac{\mathrm{p}(Q|D_i)}{\mathrm{p}(Q)}$), Figure 1 plots the distribution of these two scores for the subset of relevant documents for the 50 topics (topics 301-350) in the TREC-6 task. The likelihood scores have a very wide distribution across queries while the likelihood ratio scores are more tightly clustered. Box plots are used to indicate the score distributions. The center line in the box indicates the mean value while the lower and upper edges of the box indicate, respectively, the lower and upper quartiles. The vertical lines extending below and above the box show the entire range of the scores. We observe that the document likelihood scores can differ drastically depending on the topic. The best score for some topics (e.g., 309 and 316) are worse than the lowest scores for other topics (e.g., 315 and 339). Scoring the documents using the likelihood ratio puts the scores for the different topics on a much more comparable range. These scores can be interpreted as how much more likely the document has become after the topic is specified than before.

In the computation of the standard information retrieval measures of recall, precision, and mean average precision (mAP), each topic is treated independently. Precision-recall curves are generated for each topic separately using individual thresholds. These separate curves are then combined to create an aggregate precision-recall curve and the single number mAP measure. Since document scores are not compared across the different topics in the computation of these standard information retrieval measures, they will be identical for both the like-

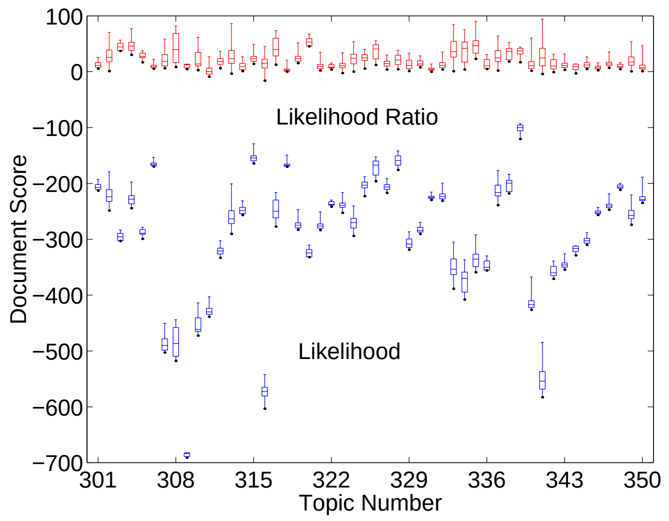

Figure 1: Distribution of likelihood and likelihood ratio scores for the relevant documents for topics 301-350 in the TREC-6 task.

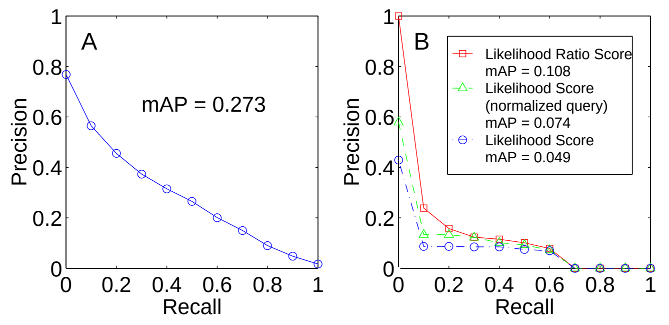

Figure 2: (A) Precision-Recall curve and mean average precision (mAP) score on the TREC-6 ad hoc task using a mixture weight of $\alpha = 0.5$. (B) Precision-Recall curves resulting from using a single threshold across all topics on the TREC-6 data for three different scoring methods.

lihood and likelihood ratio scores. In Figure 2A, we plot the resulting aggregate precision-recall curve and mean average precision (mAP) measure on the TREC-6 ad hoc task for the 50 topics (301-350). This is the baseline performance of our retrieval model using the preliminary retrieval run and a fixed topic-independent mixture weight of $\alpha = 0.5$. A performance of mAP=0.273 is achieved.

There are certain related applications, such as document clustering and topic detection, where it is important to be able to compare document scores across different "topics." To quantify how much the likelihood ratio score can help in these situations, we can generate a precision-recall curve that results from using a *single threshold* across all the different topics. In this way, we can measure the ability of the different scoring methods to handle across topic score comparisons. In Figure 2B, we show such recall-precision curves and the associated mAP measure for the 50 topics on the TREC-6 ad hoc data using three different scoring methods. As expected, the raw likelihood score performs poorly when cross topic score are compared. A normalized likelihood score (normalized by the number of the terms in the topic) gives slightly better results. However, the likelihood ratio score, which is not only normalized by the number of terms in the topic but also by the prior likelihoods of the terms, gives even better performance.

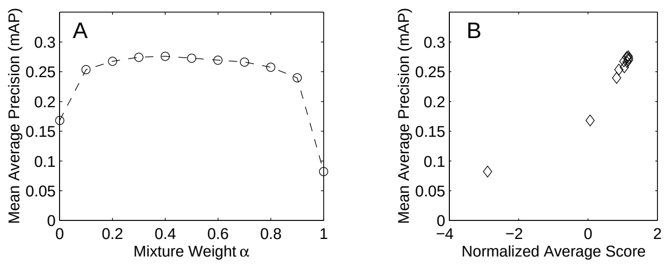

Figure 3: (A) Retrieval performance in mean average precision (mAP) on the TREC-6 task as a function of the value of the mixture weight α. (B) Scatter plot of mAP versus the normalized average score of the top documents for each of the different α weights.

Mixture Weight Estimate	mAP
Fixed ($\alpha = 0.5$)	0.273
Topic-Independent ($\alpha = 0.434$)	0.275
Topic-Dependent (variable α)	0.278

Table 4: Retrieval performance in mean average precision (mAP) on the TREC-6 task using different estimates of the mixture weight α.

3.4. Mixture Weights

In this section, we explore the effect of different α mixture weight estimates on retrieval performance and examine topic-specific and topic-independent α's. To quantify the sensitivity of the model to the mixture weight α, we explore a range of possible weight values and measure the resulting retrieval performance. In Figure 3A, we plot retrieval performance in mean average precision (mAP) on the TREC-6 ad hoc task as a function of the value of the mixture weight α. We see that although retrieval performance does vary with the value of α, there is a relatively large range of stable and good performance.

A scatter plot of mAP versus the normalized average score of the top retrieved documents for each of the different α weights is shown in Figure 3B. The plot shows that retrieval performance is well correlated ($\rho = 0.96$) with the document scores. This means that we can use the document scores to find an appropriate value of α that can be expected to give reasonably good retrieval performance. In fact, the automatic α parameter estimation procedure that we described in Section 2.1 tries to maximize the likelihood of topic Q given document D_i, $\mathrm{p}(Q|D_i)$, which is the numerator of the document score (2). Since the denominator of the score, $\mathrm{p}(Q)$, remains unchanged, this is equivalent to maximizing the entire document score. As shown in Table 4, running the preliminary retrieval pass using a fixed weight of $\alpha = 0.5$ results in a retrieval performance of mAP=0.273. Performance improves slightly to mAP=0.275 when we use the automatically estimated topic-independent weight of $\alpha = 0.434$.

Since topic statements can be very different from one another, we can expect that using the same α weight for every topic is probably suboptimal. This is indeed the case as illustrated in Figure 4, which plots retrieval performance in average precision (AP) for three different topics (327, 342, and 350) from the TREC-6 ad hoc task as a function of the value of the mixture weight α. We see that the optimal value of α for each topic can be very different. To address this issue, we can estimate topic-dependent α's, as discussed in Section 2.1. In Figure 5, we plot the distribution of the automatically estimated topic-dependent α mixture weights for the 50 topics (301-350) in the TREC-6 task. Many of the weights are centered around the topic-independent estimated value of α=0.434 but there are several topics that have weights at the extreme ends of the range. Using these topic-dependent α mixture weights, retrieval performance is further improved to mAP=0.278 as shown in the last row of Table 4.

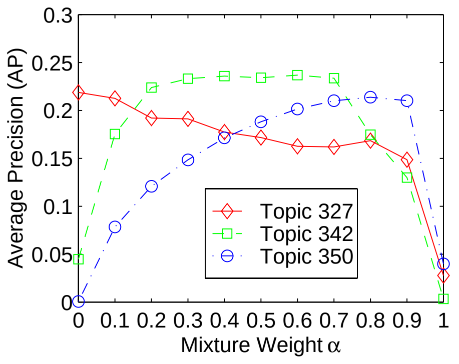

Figure 4: Retrieval performance in average precision (AP) for topics 327, 342, and 350 from the TREC-6 task as a function of the value of the mixture weight α.

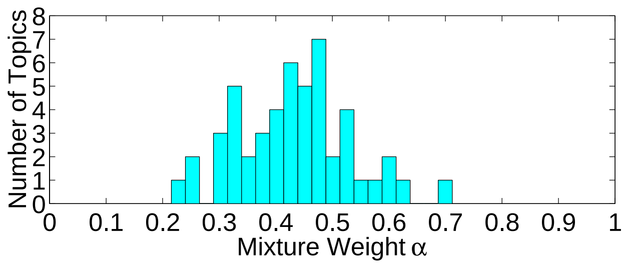

Figure 5: Distribution of the automatically estimated topic-dependent α mixture weights for topics 301-350 in the TREC-6 task. The pooled α is 0.434 and the average α is 0.432.

3.5. Automatic Feedback

In this section, we evaluate the automatic relevance feedback procedure described in Section 2.2 and examine its ability to improve retrieval performance. Recall that during the feedback process, a new topic Q' is created by removing certain terms from the original topic Q and adding new terms (with appropriate term weights) from the top scoring documents obtained from a preliminary retrieval run. The number of new terms added to Q' can be controlled by changing the threshold ϕ in the term selection criteria (42). Lowering the value of ϕ adds more terms. Note that new query terms are added in order of decreasing contribution to the total score; terms that contribute most to improving the score are added first.

Figure 6 plots retrieval performance, measured in mean average precision (mAP), on the TREC-6 ad hoc task as the number of terms in the new topic Q' is varied. Running the preliminary retrieval pass using the original topics, which average 27 unique terms each, gives a performance measure of mAP=0.273. Using automatic feedback to modify the topic results in significant performance improvements as illustrated in Figure 6. As more terms are included in the new topic Q', performance improves sharply, reaches a maximum at around 250-300 terms, declines slightly, and then levels off. The retrieval performance peaks at mAP=0.317 for approximately 250 terms.

It is interesting to note that performance is relatively stable over a wide range of topic sizes spanning 200 to 700 terms. By significantly increasing the number of terms in the topic, one may expect that the topic specification may become too broad and, as a result, the retrieval performance will be adversely affected. However, this does not happen in our case because the terms added to the new topic Q' are weighted proportionally to their score contribution as specified in (40). As a result, many of the additional terms will only have a small effect on the total score.

In terms of determining an appropriate ϕ threshold to use, one

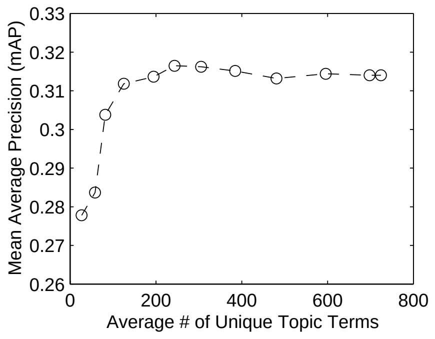

Figure 6: Retrieval performance in mean average precision (mAP) on the TREC-6 ad hoc task using the automatic feedback procedure as the number of terms in the new topic Q' is varied. By lowering the threshold ϕ in the term selection criteria (42), more terms are included in the new topic.

possibility is to simply set $\phi = 1.0$ so all terms that contribute positively to the score will be included. This corresponds to adding the maximum number of terms allowed by our procedure. Using this threshold value on the TREC-6 ad hoc task, the average number of unique terms in the new query Q' grows to 724.2. However, from the behavior shown in Figure 6, the same or even slightly better performance can be achieved by using many fewer terms. We find empirically that a reasonable threshold to use is $\phi = 0.25 \times \mathrm{S}_{\max}(D_i, Q)$, where $\mathrm{S}_{\max}(D_i, Q)$ is the score of the top retrieved document D_i for topic Q. This relative threshold value puts us in the stable performance region without adding too many terms to the new topic Q'.

We conclude that incorporating the automatic feedback processing stage into the retrieval system significantly improves retrieval performance. Large gains of 0.035 to 0.04 in absolute mean average precision (from mAP=0.278 to 0.317) are obtained.

3.6. Topic Section Weighting

As described in Section 3.1, the queries or topics statements for the retrieval tasks consist of three different sections: a title, a description, and a narrative. We can expect that the different sections contain different amounts of useful information. To quantify how useful each section is in finding the relevant documents for the topic, we can evaluate the retrieval performance resulting from using each topic section individually. In Table 5, we show retrieval performance in mean average precision (mAP) on the TREC-6 ad hoc task using the different topic sections. We examine the use of the title, description, and narrative sections individually, the title and description sections combined (T+D), and all three sections together (T+D+N). Retrieval performance after the preliminary and feedback retrieval stages are shown along with the average number of unique terms in each topic section. We can make several observations. First, the different topic sections vary greatly in their size. The title, description, and narrative sections average 2.5, 8.8, and 21.7 unique terms, respectively. Second, even though the title section contains the fewest terms, its preliminary retrieval performance is better than that of the other two sections. This implies that the terms from the title section are more useful than those from the other sections. Third, using multiple topic sections results in better performance. Combining the title and description (T+D) gives performance that is better than any of the individual sections, and using all three (T+D+N) gives even better performance. Fourth, automatic feedback improves performance in all cases but is more effective when there are more terms in the topic statement. In particular, the gain for the title section is small compared to the gains for the other sections.

In the above experiments, when we combined the different topic sections, we weighted each section equally. This means that in the

Topic Section	Avg # Unique Topic Terms	mAP	
		Preliminary	Feedback
Title (T)	2.5	0.225	0.230
Description (D)	8.8	0.178	0.221
Narrative (N)	21.7	0.218	0.253
T+D	9.5	0.247	0.296
T+D+N (All)	27.0	0.278	0.317

Table 5: Retrieval performance in mean average precision (mAP) on the TREC-6 ad hoc task using different sections of the topics: title, description, and narrative individually, title and description combined (T+D), and all three sections together (T+D+N). The second column shows the average number of unique terms in each section. The third and fourth columns show performance after the preliminary and feedback retrieval stages, respectively.

	mAP	
Topic Section	Preliminary	Feedback
T+D	0.247	0.296
T+D (weighted)	0.260	0.297
T+D+N	0.278	0.317
T+D+N (weighted)	0.303	0.325

Table 6: Retrieval performance in mean average precision (mAP) on the TREC-6 ad hoc task with and without topic section weighting. Performance is shown for two different topic configurations: title and description combined (T+D), and all three sections (title, description, and narrative) together (T+D+N). Performance after the preliminary and feedback retrieval stages are shown.

T+D+N case which combines all three sections, the title section only contributes, on average, 2.5 terms to the combined topic while the narrative section contributes 21.7 terms. From the performance of the individual topic sections in Table 5, it is clear that the terms in the title section are more useful than those in the narrative section. Maybe emphasizing terms from some sections (e.g., the title), more than terms from other sections (e.g., the narrative) in the formation of the combined topic will result in better performance than just equally weighting all the sections. This is indeed the case. In [11], they found that weighting the topic terms based on what section they are in improved retrieval performance. In [14], the output from several retrieval runs using the individual topic sections are combined to give improved performance.

We can adopt a similar approach of weighting terms based on their topic section membership to try to further improve retrieval performance. One method is to weight the terms from each topic section in proportion to the average score of the top documents retrieved using that section. The idea is that topic sections that give higher document scores should be emphasized more than those that give lower scores. We are basically using the document score as a predictor of retrieval performance which is consistent with our retrieval model which ranks documents based on descending values of the document scores. Because the scores are normalized (likelihood ratios), we are able to compare them across different topic statements (consisting of different topic sections) to determine which topic formulation is better. Basically, we run three retrieval passes using the title, description, and narrative sections individually, compute the average score of the top retrieved documents from each run, and then use those scores in weighting the terms from the different topic sections. The process used to select the set of top scoring documents is the same as the one used in the automatic feedback procedure (24). For each new task, this procedure is used to automatically determine the appropriate section weights. Using this topic section weighting scheme on the TREC-6 ad hoc task, we get section weights of 4.2 for the title, 1.8 for the description, and 1.0 for the narrative. This weighting emphasizes the title section the most, then the description section, and finally the narrative section.

Weighting the topic sections in this way results in a small but consistent performance improvement over weighting each section equally, as shown in Table 6. Retrieval performance in mean average precision (mAP) on the TREC-6 ad hoc task with and without topic section weighting is shown for two different topic configurations: title and description combined (T+D), and all three sections (title, description, and narrative) together (T+D+N). The effect of the topic section weighting is greater on the preliminary retrieval pass than on the automatic feedback pass. Recall that the feedback process already includes term selection and term weighting. As a result, some of the gains from the section weighting may already be accounted for in the feedback processing.

	mAP	
Topic Section	Preliminary	Feedback
T+D	0.212	0.243
T+D+N (All)	0.250	0.284

Table 7: Retrieval performance in mean average precision (mAP) on the TREC-7 ad hoc task using different topic specifications: title and description combined (T+D), and all three sections together (T+D+N). Performance for the preliminary and automatic feedback retrieval stages are shown.

4. INFORMATION RETRIEVAL PERFORMANCE

All of the above experiments were conducted on the TREC-6 ad hoc text retrieval task. These development experiments were used to configure the system and to tune some system parameters. The final retrieval system has the following configuration:

- Dynamic (for each query) and automatic estimation of the mixture parameter α using the procedure described in Section 2.1 with the following parameter: M=5.
- Use of the second pass automatic relevance feedback procedure described in Section 2.2 with the following parameters: γ=0.75 (Equation 24) and $\phi = 0.25 \times \mathrm{S}_{\max}(D_i, Q)$ (Equation 42), where $\mathrm{S}_{\max}(D_i, Q)$ is the score of the top retrieved document D_i for topic Q.
- Use of the query section weighting procedure described in Section 3.6 with the following parameter: γ=0.75 (Equation 24). The section weights are automatically determined for each new set of test queries.

Now that the system configuration is set, we need to evaluate the performance of the final retrieval system on new sets of held-out test data. We use the TREC-7 and TREC-8 ad hoc retrieval tasks, described in Section 3.1, for this purpose.

4.1. Retrieval Performance on the Test Set

In Table 7, we show the performance (in mAP) of our system on the TREC-7 ad hoc task. Retrieval is done using two types of topics: one consisting of the title and description sections only (T+D) and the other consisting of all three (title, description, and narrative) sections (T+D+N). Performance is shown for the preliminary retrieval pass and the automatic feedback pass. We observe that automatic feedback significantly improves performance for all conditions and that using longer topic statements is better. The performance level of mAP=0.284 on this task is competitive with the performance of the state-of-the-art retrieval systems on the identical task as reported in the TREC-7 conference [7].

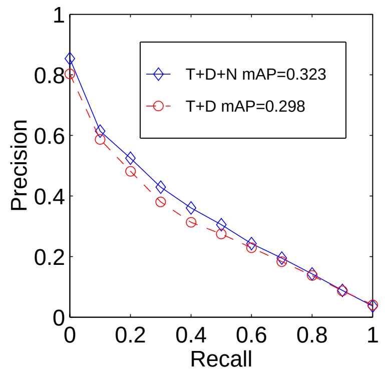

Figure 7: Precision-Recall curves for the TREC-8 ad hoc task. Performance using topics consisting of title and description (T+D), and full topics consisting of the title, description, and narrative sections (T+D+N) are shown.

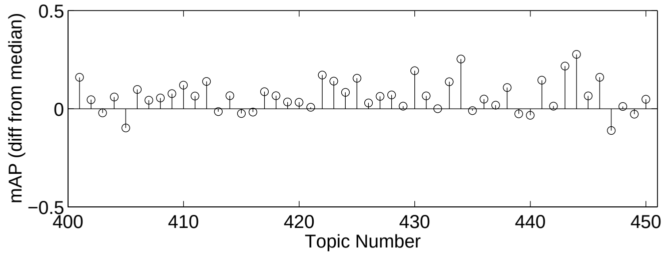

Figure 8: Difference (in mean average precision) from the median for each of the 50 topics in the TREC-8 ad hoc task. Full topics consisting of the title, description, and narrative sections are used.

4.2. Retrieval Performance on the Evaluation Set

We participated in the 1999 TREC-8 ad hoc text retrieval evaluation [8]. Performance on the official TREC-8 ad hoc task using our probabilistic retrieval model is shown in Figure 7. Two retrieval runs were submitted: one consisting of the title and description sections only (T+D) and the other consisting of all three (title, description, and narrative) sections (T+D+N). A performance of mAP=0.298 is achieved using the shorter topics; the full topics gave a mAP=0.323. Out of the 55 participating systems that used the short topic description, our system ranked sixth behind systems that had mAPs of 0.321, 0.317, 0.317, 0.306, and 0.301. Out of the 37 participating systems that used the entire topic description, our system ranked fourth behind systems that had mAPs of 0.330, 0.324, and 0.324. Difference in mAP from the median performance for each of the 50 topics for the full topic run (T+D+N) are shown in Figure 8. Of the 50 topics, 40 scored at or above the median level and seven achieved the maximum score. On this task, we again see that our retrieval model is very competitive with current state-of-the-art retrieval systems.

5. SUMMARY

In this paper we present a novel probabilistic information retrieval model and demonstrate its capability to achieve state-of-the-art performance on large standardized text collections. The retrieval model scores documents based on the relative change in the document likelihoods, expressed as the ratio of the conditional probability of the document given the query and the prior probability of the document before the query is specified. Statistical language modeling techniques are used to compute the document likelihoods and the model parameters are estimated automatically and dynamically for each query to optimize well-specified maximum likelihood objective functions. An automatic relevance feedback strategy that is specific to the probabilistic model is also developed. The procedure automatically creates a new query (based on the original query and a set of top-ranked documents from a preliminary retrieval pass) by selecting and weighting query terms so as to maximize the likelihood ratio scores of the set of documents presumed to be relevant to the query. To benchmark the performance of the new retrieval model, we use the standard ad hoc text retrieval tasks from the TREC-6 and TREC-7 text retrieval conferences. Official evaluation results on the 1999 TREC-8 ad hoc text retrieval task are also reported. Experimental results indicate that the model is able to achieve performance that is competitive with current state-of-the-art retrieval approaches.

6. ACKNOWLEDGMENTS

This research was performed under the supervision of Dr. Victor W. Zue and was supported by DARPA under contract N66001-96-C-8526 and monitored through NCCOSC.

7. REFERENCES

[1] D. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*. Cambridge, MA: Academic Press, 1982.

[2] T. Cover and J. Thomas, *Elements of Information Theory*. New York: John Wiley and Sons, 1991.

[3] F. Crestani, M. Lalmas, C. J. V. Rijsbergen, and I. Campbell, “Is this document relevant?...Probably: A survey of probabilistic models in information retrieval,” *ACM Computing Surveys*, vol. 30, no. 4, pp. 528–552, Dec. 1998.

[4] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *Journal of the Royal Statistical Society*, vol. 39, no. 1, pp. 1–38, 1977.

[5] W. A. Gale and G. Sampson, “Good-Turing frequency estimation without tears,” *Journal of Quantitative Linguistics*, no. 2, pp. 217–237, 1995.

[6] D. K. Harman, ed., *Sixth Text REtrieval Conference (TREC-6)* Gaithersburg, MD, USA, National Institute for Standards and Technology, 1997. NIST-SP 500-240.

[7] D. K. Harman, ed., *Seventh Text REtrieval Conference (TREC-7)* Gaithersburg, MD, USA, National Institute for Standards and Technology, 1998. NIST-SP 500-242.

[8] D. K. Harman, ed., *Eighth Text REtrieval Conference (TREC-8)* Gaithersburg, MD, USA, National Institute for Standards and Technology, 1999. NIST-SP.

[9] D. Hiemstra and W. Kraaij, “Twenty-one at TREC-7: Ad-hoc and cross-language track,” in *Seventh Text REtrieval Conference (TREC-7)*, Gaithersburg, MD, USA, 1998. NIST-SP 500-242.

[10] F. Jelinek, *Statistical Methods for Speech Recognition*. Cambridge, MA: MIT Press, January 1999.

[11] D. Miller, T. Leek, and R. Schwartz, “BBN at TREC-7: Using hidden markov models for information retrieval,” in *Seventh Text REtrieval Conference (TREC-7)*, Gaithersburg, MD, USA, 1998. NIST-SP 500-242.

[12] J. Ponte and W. B. Croft, “A language modeling approach to information retrieval,” in *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*,, pp. 275–281, 1998.

[13] M. Porter, “An algorithm for suffix stripping,” *Program*, vol. 14, no. 3, pp. 130–137, July 1980.

[14] S. Robertson, S. Walker, and M. Beaulieu, “Okapi at TREC-7: automatic ad hoc, filtering, VLC and interactive,” in *Seventh Text REtrieval Conference (TREC-7)*, Gaithersburg, MD, USA, 1998. NIST-SP 500-242.

[15] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*. New York: McGraw-Hill, 1983.