Indri at TREC 2006: Lessons Learned From Three Terabyte Tracks

Donald Metzler, Trevor Strohman, W. B. Croft

Center for Intelligent Information Retrieval
University of Massachusetts, Amherst

Abstract

This report describes the lessons learned using the Indri search system during the 2004-2006 TREC Terabyte Tracks. We provide an overview of Indri, and, for the *ad hoc* and named page finding tasks, discuss our general approach to the problem, what worked, what did not work, and what could possibly work in the future.

1 Introduction

The Terabyte Track consists of three tasks, which include the efficiency task, the ad hoc retrieval task, and the named page finding task. In our previous TREC reports, we only describe our official submissions [7, 8]. However, many additional methods were tried. Some of these methods produced poor results or failed to improve effectiveness. Others showed promise, but were never fully investigated due to time constraints. In this report, we first provide an overview of the Indri retrieval system [10] and then summarize the outcomes of our experiments using Indri by describing those methods what worked, those that did not, and those that potentially could work for the *ad hoc* and named page finding tasks.

2 System Overview

The Indri retrieval system was built to evaluate complex structured queries on large corpora. Development began in October 2003, and the system was finished just in time to run experiments for the first Terabyte Track in 2004.

Indri was originally meant to be a small modification to the Lemur Project code. There were enough scalability issues with the original Lemur code that we built almost an entirely new system. The Indri system is now distributed as a component of the new, larger Lemur toolkit[1]. It is open source and freely available for download.

Our first goal with Indri was to create a platform for experimenting with ranking strategies for large collections. We worked to balance the competing goals of efficiency, effectiveness and flexibility. Our requirements for flexibility led us to adopt a document-at-a-time scoring strategy. This strategy allowed us to experiment with large collections and comple queries on systems with very little memory. We also chose to build indexes with position information, pseudo-relevance feedback structures, and a compressed copy of the collection built-in. These large indexes enabled us to quickly experiment with the retrieval models outlined in this paper. These choices meant that our system was not particularly interesting in the efficiency task, especially in 2005 and 2006 as other teams began to hone the efficiency of their systems.

We added multithreading to our system in 2005, which significantly improved our query performance. Our own experiments showed impressive improvements in speed with an additional thread on a single CPU system, and a near doubling in throughput in distributed mode. However, in the 2005 efficiency track we were not allowed to run multiple queries simultaenously in official runs. We did not participate in the 2006 efficiency track, which allowed for parallelism with multiple query streams.

[1] http://www.lemurproject.org/indri

Indri was built from the start to support querying across a cluster of machines, with results that are guaranteed to be the same as if all the documents were stored in a single index. We achieve this by sharing collection statistics with all nodes in the cluster. The first phase of each query is a collection statistics phase, where the query broker requests statistics from each of the query processing nodes. The query is then processed using the gathered statistics. When processing queries sequentially, a 6-node cluster achieved about three times the query throughput of a single machine. By using two threads, the same 6-node cluster achieved about 4.5 times the query throughput of a single machine.

3 Ad Hoc Task

For the *ad hoc* retrieval task, our strategy was to get the most out of the Indri query language as possible. The query language provides support for term proximity operators, such as ordered and unordered windows, synonyms, matching based on document structure constraints, document priors, and the ability to assign weights to various query language constructs. Therefore, our goal was to find the best way of automatically transforming a TREC topic into a complex structured Indri query. We now briefly describe the lessons learned from the various formulations and strategies that we tried.

3.1 What Worked

3.1.1 Term Proximity

One of the most significant and consistent improvements in effectiveness that we observed came from using term proximity operators. However, as will become apparent shortly, blind application of term proximity operators does not work particularly well. Instead, we found that one specific term proximity formulation, Metzler's dependence model formulation, consistently improved effectiveness over a bag of words baseline [6].

Given a query, the dependence model is essentially a feature expansion mechanism. The original query is expanded to include exact phrase (`#1`) and unordered window (`#uwN`) features. Two very important parts of this formulation, which are often overlooked or not present in similar models, are feature weighting and the feature smoothing. Feature weights are learned by directly maximizing mean average precision via hill-climbing. For feature smoothing, we found that it is valuable to apply different amounts of smoothing to single term features and proximity features [5].

The results in Table 1 compare our term proximity formulation (DM-LM) with a standard bag of words language modeling-based approach (QL). For the entire set of Terabyte Track topics, the term proximity formulation outperforms the bag of words approach by 8.2% in terms of mean average precision.

3.1.2 Query Expansion

We also found query expansion to be another valuable strategy. For query expansion purposes, we use a technique that generalizes Lavrenko's relevance models [4] to work with the useful term proximity features described in the previous section.

We found that query expansion on top of a bag of words model helped significantly. However, when it was was used on top of a strong term proximity formulation, the improvement was amplified. That is, we observed an additive effect, rather than a dampening one.

We provide results in Table 1 for two query expansion techniques that are built upon the dependence model framework. The QE-LM approach uses language modeling features, as described in [6]. The QL-BM25 approach uses analagous BM25 features. Interestingly, the LM features seem to outperform the BM25 features on the 2004 and 2005 topics, but not the 2006 topics. More analysis must be done to understand this phenomenon better.

3.1.3 Document Quality Prior

Finally, in terms of what helped, we saw mixed results when using a document quality prior. The prior helped when used in conjunction with a bag of words approach, but actually hurt when used with a more complex formulation that used term proximity and query expansion.

The document quality prior is based on two features that aim to measure how likely the document is to contain useful text content. It was meant to significantly penalize documents that only consist of tables, java applets, and

	QL(T)	DM-LM(T)	QE-LM(T)	QE-BM25(T)	QE-LM(TDN)
2004 Topics (701-750)	0.2870	0.3067	0.3326	0.3216	0.3650
2005 Topics (751-800)	0.3432	0.3632	0.4002	0.3878	0.4287
2006 Topics (801-850)	0.3071	0.3444	0.3452	0.3687	0.4252
All Topics (701-850)	0.3126	0.3383	0.3595	0.3596	0.4065

Table 1: Ad hoc task mean average precision values for various retrieval strategies. QL represents query likelihood, DM-LM is dependence models with language modeling features, QE-LM is dependence model query expansion using LM features, and QE-BM25 is dependendence model query expansion using BM25 features. In addition, T indicates a run that only makes use of the title field of the TREC topic, while TDN indicates a run that makes use of the title, description, and narrative fields. All runs are automatic.

images, as these documents are unlikely to be relevant to *ad hoc* query requests. See [8] and [11] for more details.

3.2 What Did Not Work

3.2.1 Statistical Phrases and WordNet

Although improvements were achieved using term proximity features, as described previously, it took a great deal of effort and experimentation to get to that point. The first set of failed experiments focused primarily on statistical phrases and WordNet.

A statistical phrase dictionary was built. Then, given a query, if any subphrase within the query was found in the statistical phrase dictionary it was added as an exact phrase to the query. Various formulations were tried, including removing original query terms if they occured in a statistical phrase, trading off weight between query terms and statistical phrases, among others. However, none of the experiments that were tried improved upon the bag of words baseline. Similar experiments were carried out using WordNet to automatically construct synonyms of terms and phrases. Such formulations performed even worse than the ones done using statistical phrases.

3.2.2 Document Structure

As observed at past TREC Web Tracks [2], we found no use for document structure. Various formulations were tried, such as our named page formulation (described below), but with no success. In all of our experiments, the optimal parameter settings found gave zero weight to all fields except the main body of the document.

3.3 What Could Work

One promising area of future work includes using an external resource of query expansion terms. It was recently shown that using the web as an external resource significantly improves *ad hoc* retrieval effectiveness on the WT10g web collection [1].

Several preliminary experiments were done on the GOV2 collection and found to significantly improve effectiveness when compared to using GOV2 as the source of query expansion terms. However, due to time constraints, no further experiments were done. It is likely that a combination of term proximity and query expansion against the web will yield even better effectiveness than that achieved in Table 1.

4 Named Page Finding Task

For the named page finding task, our strategy is to construct queries that utilize Indri's document structure and document prior probability capabilities.

4.1 What Worked

4.1.1 Structure, Link Analysis, Priors

We found that the *de facto* best-practice named page finding formulation, which includes using document structure [9], link analysis, and document priors [3], continued to work well on the 2005 and 2006 named paged finding topics. Our system made use of all three techniques.

	QL-MM	DM-MM
2005 Topics (601-872)	0.4143	0.4405
2006 Topics (901-1081)	0.4980	0.5123
All Topics	0.4493	0.4705

Table 2: Named page finding task mean reciprocal rank values for various retrieval strategies. Here, QL-MM represents a mixture of unigram language models approach and DM-MM represents a mixture of dependence models approach. In both approaches, the mixing models are estimated from the `title`, `heading`, `anchor`, and `mainbody` representations of a web page. In addition, inlink and PageRank information are incorporated in the form of a document prior probability in both models.

4.1.2 Term Proximity

As with the *ad hoc* task, we also found term proximity models to be effective for the named page finding task. Although the two tasks are fundamentally different, it is interesting to see that term proximity improves both.

Table 2 shows our named page finding results from 2005 and 2006. For this year's track, we used the same general query formulation as last year. For more details see [8]. The table compares a formulation that makes use of a mixture of unigram language models with one that uses a mixture of term dependence models. Both formulations also include document priors. As the results show, the formulation that uses term proximity improves MRR by 4.7% over the unigram formulation.

4.2 What Did Not Work

Our primary focus was the *ad hoc* task, and therefore we did not investigate many alternative query formulations for the named page finding task. For this reason, there was no single method that we tried that did not work.

4.3 What Could Work

Some preliminary experiments were done using multiple-Bernoulli language models in place of multinomial models. The rationale here is that the `title` field often contains very little, if any, text, and therefore term frequency, which is an important aspect of the multinomial model, is much less important, and may be modeled better by a multiple-Bernoulli distribution. This is an interesting area of future work.

Acknowledgments

This work was supported in part by the CIIR, in part by NSF grant #CNS-0454018 , and in part by ARDA and NSF grant #CCF-0205575. Any opinions, findings and conclusions or recommendations expressed in this material are the author's and do not necessarily reflect those of the sponsor.

References

[1] Fernando Diaz and Donald Metzler. Improving the estimation of relevance models using large external corpora. In *Proceedings of SIGIR 2006*, pages 154–161, 2006.

[2] David Hawking and Nick Craswell. Overview of the trec-2001 web track. In *Proceedings of TREC 2001*, 2001.

[3] Wessel Kraaij, Thijs Westerveld, and Djoerd Hiemstra. The importance of prior probabilities for entry page search. In *Proceedings of SIGIR 2002*, pages 27–34, 2002.

[4] Victor Lavrenko and W. Bruce Croft. Relevance-based language models. In *Proceedings of SIGIR 2001*, pages 120–127, 2001.

[5] Donald Metzler and W. Bruce Croft. Combining the language model and inference network approaches to retrieval. *Information Processing and Management*, 40(5):735–750, 2004.

[6] Donald Metzler and W. Bruce Croft. A markov random field model for term dependencies. In *Proceedings of SIGIR 2005*, pages 472–479, 2005.

[7] Donald Metzler, Trevor Strohman, Howard Turtle, and W. Bruce Croft. Indri at trec 2004: Terabyte track. In *Proceedings TREC 2004*, 2004.

[8] Donald Metzler, Trevor Strohman, Yun Zhou, and W. Bruce Croft. Indri at trec 2005: Terabyte track. In *Proceedings TREC 2005*, 2005.

[9] Paul Ogilvie and Jamie Callan. Combining document representations for known-item search. In *Proceedings SIGIR 2003*, pages 143–150, 2003.

[10] Trevor Strohman, Donald Metzler, Howard Turtle, and W. B. Croft. Indri: A language model-based serach engine for complex queries. In *Proceedings of the International Conference on Intelligence Analysis*, 2004.

[11] Yun Zhou and W. Bruce Croft. Document quality models for web ad hoc retrieval. In *Proceedings of CIKM 2005 (to appear)*, 2005.