Indri at TREC 2005: Terabyte Track

Donald Metzler, Trevor Strohman, Yun Zhou, W. B. Croft

Center for Intelligent Information Retrieval
University of Massachusetts, Amherst

Abstract

This work details the experiments carried out using the Indri search engine during the TREC 2005 Terabyte Track. Results are presented for each of the three tasks, including efficiency, *ad hoc*, and named page finding. Our efficiency runs focused on query optimization techniques, our *ad hoc* runs look at the importance of term proximity and document quality, and our named-page finding runs investigate the use of document priors and document structure.

1 Introduction

This year, the Terabyte Track was expanded from a single *ad hoc* task to include an efficiency and named-page finding task. These new tasks provided new and interesting environments in which to test our search engine, Indri[1] [13]. As evidenced by last year's Terabyte Track results, Indri is a highly efficient, highly effective search engine. The underlying retrieval model is based on a combination of the inference network [15] and language modeling [3] approaches to retrieval [7]. Indri supports a robust query language, based on the InQuery query language [1]. Our primary goal this year was to further tweak our system both in terms of efficiency and effectiveness in large scale retrieval settings.

The remainder of this paper details the experiments and results obtained for each of the three Terabyte Track tasks.

[1] Available for download at: http://lemurproject.org/indri/

2 Efficiency Task

Indri was a brand new retrieval system at last year's TREC conference. In the past year, we have added to its capabilities and have attempted to improve its speed without sacrificing retrieval effectiveness. To this end, Indri includes new query processing techniques [14] and a new multithreaded architecture [12].

In contrast to last year's system, Indri is now multithreaded, which allows many queries to be processed at once. Unfortunately, the rules specifically prohibited this optimization this year, so our official runs do not include this optimization. We report threaded results in this paper for comparison.

The indexing code for the system is conceptually similar to last year's system, although it has been completely rewritten. The main feature of the new code is that it is concurrent; queries can be processed while documents are being added to the system, and each document is available for query immediately after it is added, with no document batching necessary. We also separated the vocabulary structures into two B-Trees; one for frequent terms (those appearing more than 1000 times in the collection) and one for infrequent terms. The frequent terms tree is small enough that it is quickly cached into memory by the operating system; future lookups to this structure incur no disk I/O. Since most query terms come from the frequent terms list, disk I/O is reserved for the inverted lists.

Our goal this year was to try to make our current system fast while not sacrificing any of its capability. To this end, our index structures include inverted lists, direct lists, and a compressed version of

CPU	Intel Pentium 4 2.6GHz × 1
Bus speed	800MHz
OS	Linux 2.6.10 (Fedora Core 3)
Memory	2GB
Boot volume	Western Digital 40G (WD400EB-75CPF0)
Work volume	Western Digital 250GB × 3 (WD2500JB-00EVA0) RAID 0 Average write seek: 10.9ms Average read seek: 8.9ms Rotational speed: 7200rpm
Network	1Gb/s Ethernet (Intel 82540EM)

Table 1: Machine configuration. Six machines at a total cost of $9000 USD, purchased in early 2004.

Run ID	Total (s)	Avg (s)
indri05Eql	71700	1.43
indri05EqlD	24720	0.49

Table 2: Summary of official efficiency task runs.

the document collection. The query code and index structures are the same as those used for the effectiveness tasks, although we used more complex query formulations in the effectiveness task.

All of our machines meet the specifications in Table 1. For the first run, we used a single machine (2079 mins. to build index), while for the distributed run, we used 6 of these machines in parallel (327 mins. to build index) for both indexing and retrieval. The distribution of the text for indexing was done manually, but the query processing distribution was automatic.

Tables 2 and 3 show our official (unthreaded) and unofficial (threaded) runs, respectively. The tables provide total query processing and the average time across the 50,000 efficiency queries. Distributing the index, as expected, improves our official average query time from 1.43 to 0.49s per query. We also see from our unofficial numbers that threading the query director in a fully distributed environment further improves throughput by a factor of 2-3.

Threads	Setup	Total (s)	Avg (s)
2	Single	65640	1.31
2	Distributed	14520	0.29
4	Distributed	9780	0.20
8	Distributed	9000	0.18
128	Distributed	8820	0.18

Table 3: Summary of unofficial, threaded efficiency task runs.

3 Ad Hoc Task

For the *ad hoc* retrieval task this year we extended and improved the methods we investigated during TREC 2004 [9]. We are interested in studying how traditional *ad hoc* retrieval models and methods scale to large, noisy web collections, and what new types of features may exist that can be exploited in such an environment. Therefore, this year we explored the use of term proximity information, pseudo-relevance feedback, and the use of a document quality prior.

All of our *ad hoc* runs are against the entire GOV2 collection, with no special document or link structure indexed. All documents are stemmed using the Porter stemmer. Once again this year, we do not stop documents at index time. Instead, we perform query-side stopping using a list of 418 common terms. We use Bayesian smoothing, and allow single term and proximity features (i.e. `#1`, `#uw8`) to be smoothed differently. All of our runs are automatic and all the system parameters are tuned on the TREC 2004 Terabyte Track topics (701-750).

3.1 Baseline

Our baseline run this year, `indri05Aql`, is a simple title-only query likelihood run. For example, topic 758, "embryonic stem cells", is converted into the following Indri query:

```
#combine( embryonic stem cells )
```

which produces results rank-equivalent to a simple query likelihood language modeling run. We consider this to be a strong, reasonable baseline.

3.2 Dependence Model

At least year's Terabyte Track, we developed a novel mechanism for modeling term proximity features [9]. Since then, the model has been expanded and expressed in terms of a term dependence model [8]. The use of term proximity information is not new [2], but has never caught on. However, we feel it plays an increasingly important role in retrieval as collections get larger and noisier.

The model encodes the following two assumptions: 1) *the order of query terms is important*, and 2) *query terms will occur within closer proximity to each other in relevant documents*. The full details of the model are not given here due to space constraints. See [8] for a full treatment.

To give an idea of how the dependence model translates queries into Indri structured queries we give the following example, again for topic 758:

```
#weight(0.8 #combine(embryonic stem cells)
        0.1 #combine(#1(stem cells)
                     #1(embryonic stem)
                     #1(embryonic stem cells))
        0.1 #combine(#uw8(stem cells)
                     #uw8(embryonic cells)
                     #uw8(embryonic stem)
                     #uw12(embryonic stem cells)))
```

Our `indri05Adm`, `indri05AdmfS`, and `indri05AdmfL` runs employ this model.

3.3 Pseudo-Relevance Feedback

For pseudo-relevance feedback, we use a modified version of Lavrenko's relevance model [5]. Given an initial query Q_{orig}, we retrieve a set of N documents and form a relevance model from them. We then form Q_{RM} by wrapping a `#combine` around the k most likely terms from the relevance model that are not stopwords. Finally, an expanded query is formed that has the following form:

`#weight(` $\lambda_{fb}\ Q_{orig}\ (1.0 - \lambda_{fb})\ Q_{RM}$)

Last year we were disappointed with our pseudo-relevance feedback results, which showed little improvement over the baseline. We found the parameters we had tuned on the WT10g collection were ill-suited for the GOV2 collection. As we will show, the parameters found by tuning on last year's topic set proved to yield better results this year.

Two of our runs made use of this technique. First, run `indri05AdmfS`, a title-only run, uses a dependence model query for Q_{orig}, $N = 10$, $k = 50$, and $\lambda_{fb} = 0.5$.

Next, run `indri05AdmfL`, a title + description + narrative run, uses a dependence model query created from only the title field combined with the description and narrative portions of the topic for Q_{orig}, $N = 10$, $k = 50$, and $\lambda_{fb} = 0.6$.

Current and past results indicate that the effectiveness boost from pseudo-relevance feedback techniques are amplified when used in combination with the precision-enhancing dependence model. Therefore, such a combination provides a highly effective retrieval mechanism.

3.4 Document Quality Prior

Lastly, we incorporate the notion of document quality into our retrieval models in the form of a prior probability. To achieve this, we must identify document features that are predictive of quality. We focus on two features, information-to-noise ratio and collection-document distance.

Information-to-noise ratio is simply defined as the total number of terms in the documents after indexing divided by the raw size of the document. This metric has been used with some success previously [17]. The other feature, collection-document distance, is the KL-divergence between the collection and document model.

The intuition behind this feature comes from the observation that documents, such as tables or lists, are unlikely to be relevant for *ad hoc* queries because relevant documents typically explain or describe some topic using well-formed English sentences. Therefore, if a document differs significantly from the collection model, the quality of this document may be low. The higher the CDD is, the more unusual the word distribution of the document is, and the more likely,

according to our hypothesis, that the document is of low quality.

The quality of a document is determined using a naive Bayes classifier over the two features mentioned above, with the classes defined as high and low quality. The class priors are estimated from the training data and the class conditional distributions are estimated using a non-parametric density estimation technique with a Gaussian kernel. See [16] for more details on the training data and estimation details.

The document quality prior was applied to the `indri05AdmfL` run.

3.5 Results

The results from our official runs[2] are given in Table 5. The table includes mean average precision (MAP), precision at 10 (P@10), the term smoothing parameter (μ), and the proximity feature smoothing parameter (μ_{prox}).

We see that once again this year the dependence model improves effectiveness. This year it (`indri05Adm`) provided a 7.8%, statistically significant, increase in MAP over the baseline. Last year the increase was 6.8%. This provides further evidence of the importance of term proximity features, especially for this task.

Our best title-only run (`indri05AdmfS`), which combined the dependence model with pseudo-relevance feedback, showed a 19.5% increase in MAP over the baseline and a 10.9% increase over the dependence model only run, both of which are statistically significant. Therefore, we see that pseudo-relevance feedback was much more effective this year than last, mostly due to better parameter tuning.

Finally, the title + description + narrative run that made use of dependence modeling, pseudo-relevance feedback, and the document quality prior (`indri05AdmfL`) yielded our best overall MAP. Despite this, the 4.0% improvement over the title-only version of the run was not statistically significant. Interestingly, if we perform the same run without applying the document quality prior we achieve a MAP of 0.4150, which is statistically better. Further analysis must be done to determine why the document quality prior hurt performance. We also continue to look at better ways of incorporating the information from the description and narrative fields, because we feel it should translate into even better effectiveness numbers than we are currently seeing.

[2]The Indri queries and parameter files used for our official runs are available for download at http://ciir.cs.umass.edu/~metzler/indri-tb05.tgz

Run ID	μ	μ_{prox}	MAP	P@10
indri05Aql	1500	-	0.3252	0.5840
indri05Adm	1500	4000	0.3505	0.5960
indri05AdmfS	1500	4000	0.3886	0.6340
indri05AdmfL	2000	2000	0.4041	0.6580

Table 5: Summary of official *ad hoc* task runs.

Therefore, our runs this year show that term proximity information and pseudo-relevance feedback can significantly improve *ad hoc* retrieval results on a large web collection. Results using the description/narrative fields and the document quality prior are mixed, but potential areas of future interest.

4 Named Page Finding Task

Since this was the first year that UMass participated in any type of named-page finding task, our runs were highly experimental and designed to give us a better feel for the task. Following what has been successful in the past, we used both document structure and link analysis techniques.

We indexed `title`, `mainbody`, `heading`, and `inlink` fields, where `mainbody` is the main text of the document and `inlink` consists of all the anchor text pointing to the document, if any. We also investigated the use of a number of query independent document priors, such as PageRank, inlink count, url depth, and document length. Individual priors were estimated based on empirical distributions from TREC 9 and 10 relevance judgments [4]. After some preliminary experimentation, we decided to limit our focus to PageRank and inlink count priors.

We focused on two primary named-page finding models, both of which incorporate document structure and query independent features.

Topic	Query	ql	dm	% change
786	yew trees	0.4108	0.3311	-19.4%
799	animals alzheimers research	0.1560	0.1301	-16.6%
798	massachusetts textile mills	0.1535	0.1297	-15.5%
773	pennsylvania slot machine gambling	0.5633	0.4858	-13.8%
765	ephedra ma huang deaths	0.4653	0.4044	-13.1%
	...			
792	social security means test	0.1232	0.1940	57.5%
785	ivory billed woodpecker	0.3924	0.7051	79.7%
778	golden ratio	0.0981	0.2302	135%
794	pet therapy	0.0650	0.3755	478%
769	kroll associates employees	0.0132	0.0955	623%

Table 4: Comparison of average precision using the query likelihood (ql) baseline and the dependence model formulation (dm). The topics that are the most helped and harmed, in terms of relative change in average precision, are shown.

4.1 Feature-Based Model

The first is a linear, feature-based model based on the work of Nallapati [10]. The scoring function has the following form:

$$S(D;Q) \quad = \quad \mathbf{w}^T \mathbf{f}(D,Q)$$

where $\mathbf{f}(\cdot,\cdot)$ is a feature function that maps query/document pairs into some d dimensional space and $\mathbf{w}$ is a weight vector. Given a set of training data (feature vectors plus relevance judgments), we aim to estimate the $\mathbf{w}$ that optimizes some metric. Rather than maximizing the likelihood or margin, as was done in previous studies, we directly maximize with respect to mean reciprocal rank using a simple hill climbing approach [6].

For our specific instantiation of this model, we used the 6 features used in [10] for each of the four document structure fields indexed, a PageRank feature, and an inlink count feature, for a total of 26 features. Our `indri05Nf` run makes use of this model.

4.2 Mixture of Language Models

The second model combines a mixture of language models approach with document priors [4, 11]. The following scoring function is used to rank documents:

$$P(D|Q,\Theta) \quad \propto \quad P(D) \prod_{q \in Q} \sum_i P(i) P(q|D,i)$$

where $P(i)$ are mixture weights, $P(q|D,i)$ represents language model component i, and $P(D)$ is a document prior. We use one mixture component for each indexed field, estimate $P(D)$ using PageRank and inlink count, and hand tune the mixture weights.

Both the `indri05Nmp` and `indri05Nmpsd` runs make use of this formulation, with the latter using the sequential dependence variant of Metzler's dependence model [8]. Figure 1 shows an example named page finding Indri query language formulation.

4.3 Results

The results from our official runs are given in Table 6. As we see, the results are much lower than most of the named page finding results obtained at the 2004 Web Track. It is not clear whether or not this is caused by a fault in our formulation or if it is an artefact of the data set (documents or queries).

However, when comparing our results, we see that the feature-based model (`indri05Nf`) performs the worst, which is most likely caused by the fact that we trained on the WT10g named-page finding topics, which may be inappropriate. During training, the

```
#weight( 0.1 #weight( 0.6 #prior(pagerank)
                      0.4 #prior(inlinks) )
         1.0 #weight( 0.9 #combine( #wsum( 1.0 stellwagen.(inlink)
                                           1.0 stellwagen.(title)
                                           3.0 stellwagen.(mainbody)
                                           1.0 stellwagen.(heading) )
                                    #wsum( 1.0 bank.(inlink)
                                           1.0 bank.(title)
                                           3.0 bank.(mainbody)
                                           1.0 bank.(heading) ) )
                      0.1 #combine( #wsum( 1.0 #uw8( stellwagen bank ).(inlink)
                                           1.0 #uw8( stellwagen bank ).(title)
                                           3.0 #uw8( stellwagen bank ).(mainbody)
                                           1.0 #uw8( stellwagen bank ).(heading) ) ) ) )
```

Figure 1: Indri query formulation for topic NP604, "stellwagen bank", used for the `indri05Nmpsd` run.

Run ID	MRR	S@10	Not Found
indri05Nf	0.375	0.528	0.187
indri05Nmp	0.414	0.563	0.175
indri05Nmpsd	0.441	0.583	0.171

Table 6: Summary of official named page finding task runs.

MRR values achieved were comparable to past state of the art systems. Further tests are necessary to determine why it performed so poorly.

The two mixture of language model runs (`indri05Nmp` and `indri05Nmpsd`) performed better than the feature-based model. In fact, the run that made use of the dependence model was our best run, improving the MRR of 30% of the queries and degrading 10%. This indicates that term proximity information may also be important for named page finding, as well.

Since this was our first year taking part in a named page finding task, we hope to perform a detailed failure analysis to better understand the strengths and weaknesses of our approaches. In particular, we are interested in understanding how to make better use of document priors and what potential impact term proximity information, via use of the dependence model, can help for this task.

5 Conclusions

This year we continued to investigate how to efficiently and effectively use the Indri search engine for retrieval tasks on large scale web collections. Positive results were obtained using pseudo-relevance feedback and dependence modeling for the *ad hoc* task, while using document structure and link analysis techniques proved effective for the named page finding task.

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval, in part by NSF grant #CNS-0454018, in part by Advanced Research and Development Activity and NSF grant #CCF-0205575, and in part by NSF grant #IIS-0527159. 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] James P. Callan, W. Bruce Croft, and Stephen M. Harding. The INQUERY retrieval system. In *Proceedings of DEXA-92*, pages 78–83, 1992.

[2] Charles Clarke, Gordon Cormack, and Forbes Burkowski. Shortest substring ranking (multitext experiments for trec-4). In *Proceedings of the TREC-4*, 1995.

[3] W. Bruce Croft and John Lafferty. *Language Modeling for Information Retrieval*. Kluwer Academic Publishers, 2003.

[4] 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.

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

[6] Donald Metzler. Direct maximization of rank-based metrics. Technical report, University of Massachusetts, Amherst, 2005.

[7] 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.

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

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

[10] Ramesh Nallapati. Discriminative models for information retrieval. In *Proceedings of SIGIR 2004*, pages 64–71, 2004.

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

[12] Trevor Strohman. Dynamic collections in indri. Technical report, Center for Intelligent Information Retrieval, 2005.

[13] 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.

[14] Trevor Strohman, Howard Turtle, and W. Bruce Croft. Optimization strategies for complex queries. In *Proceedings of SIGIR 2005*, pages 219–225, 2005.

[15] Howard Turtle and W. Bruce Croft. Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems*, 9(3):187–222, 1991.

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

[17] X. Zhu and S. Gauch. Incorporating quality metrics in centralized/distributed information retrieval on the world wide web. In *Proceedings of SIGIR 2000*, pages 288–295, 2000.