Indri: A language-model based search engine for complex queries[1]

Trevor Strohman, Donald Metzler, Howard Turtle and W. Bruce Croft
Center for Intelligence Information Retrieval
University of Massachusetts Amherst
Amherst, MA, 01003, USA
strohman@cs.umass.edu

Keywords: Search and Retrieval, Question Answering

1. Introduction

Search and detection technology has been a focus of DARPA and ARDA research programs since the TIPSTER program began in the early 1990s (Harman 1992). A number of innovations have been developed in this research, resulting in very significant improvements in the effectiveness of search tools. The Inquery search engine (Callan *et al*. 1995), developed at the University of Massachusetts for the TIPSTER project, provided a query language capable of representing complex queries in a probabilistic framework and was used in a number of government and commercial applications.

It has been over a decade since Inquery was developed, and in that time information retrieval has changed significantly. The information retrieval landscape now includes question answering, information extraction, language modeling, and multi-terabyte document collections. As a part of the ARDA-sponsored Lemur project, Indri was developed to respond to these new challenges.[2]

2. Query Language and Retrieval Model

2.1. Retrieval model

The retrieval model implemented in the Indri search engine is an enhanced version of the model described in (Metzler *et al.* 2004a), which combines the language modeling (Ponte and Croft 1998) and inference network (Turtle and Croft 1991) approaches to information retrieval. The resulting model allows structured queries similar to those used in Inquery to be evaluated using language modeling estimates within the network, rather than *tf.idf* estimates. Unlike many language modeling formulations, the Indri retrieval model explicitly allows features other than simple word occurrences to be used in retrieval. This allows the Indri query language to include the kinds of complex operators that Inquery included, as well as many new operators.

2.2. Query language

Text is rarely completely unstructured in practice; many documents contain dates, titles, and author fields. Even when documents do not contain explicit structure, information extraction tools can be used to identify implicit document structure, finding proper names and dates, or marking the parts of speech used in text. Indri is able to understand document structure so that it can be referenced in queries.

While all retrieval systems can search for sets of terms, the Inquery language allowed users to request that terms be found close together in a document, or in a particular order. Inquery also allowed the user to indicate the importance of terms and expressions via weights. These expressions could be used together to form precise queries. Indri builds on the Inquery tradition by including the most-used operators from the Inquery language, while adding new operators relevant for referencing document structure.

Indri can search for a word appearing in a particular field, such as 'Iraq' appearing in a title field. Indri can also search for the title itself, instead of a whole document that contains a title.

In some tasks (e.g. question answering), users may want to search for text that matches a particular lexical pattern. For instance, if a user is interested in a list of ambassador's names and the countries they represent, the following query could be used:

```
#uw5( ambassador #any:country
      #any:person )
```

This query requests documents where the word 'ambassador' appears within 5 words of the name of the name of a person and the name of a country. Queries like this can leverage the work done by information extraction sys-

[1] A longer version of this paper is available from http://ciir.cs.umass.edu as publication IR-407.
[2] Indri is available for download at http://www.lemurproject.org

tems to find more exact matches to information needs.

While Indri is capable of these kinds of structural queries, it is also meant to be a state-of-the-art topical search engine. The structural elements can be combined with topical elements in the same query, as in this one:

```
#combine( iraq iran #1(north korea)
 communication #any:person
 #date:between(1-JAN-2000 1-SEP-2001))
```

This query indicates that the user is looking for communication involving some person, related to Iraq, Iran or North Korea, between the beginning of 2000 and September of 2001. Since Indri is a probabilistic system, it searches for the best match for this query, not just for documents that meet all the criteria. For instance, a document that referenced a communication involving Iraq, Iran and North Korea is likely to be ranked highly, even if it does not fall in the date range specified. This allows users to enter highly detailed queries without the risk that important documents that do not exactly match the query will not be found.

While it is possible for trained users to use the Indri query language directly, we expect that most users will use the system through some kind of intermediate interface. Interfaces to Indri can be made domain-specific in order to help users in specific domains use Indri most effectively.

3. System Architecture

Indri supports concurrent indexing and retrieval of large collections. We have tested Indri with collections of close to a terabyte, although it has been designed to handle much larger collection sizes. If a collection is too large for one machine to process efficiently, Indri is capable of running queries using clusters of servers. Indri is written in C++, and has been tested on Linux, Solaris, Windows, and Mac OS X. We expect that it should be portable to most Unix variants with little additional work.

Indri uses a flexible parsing architecture that is capable of processing plain text, HTML, XML, and PDF documents. On Windows machines, it can also process Word and PowerPoint documents. In all cases, Indri is capable of dealing with Unicode text in UTF-8 encoding. As many important documents are not stored in one of these formats, Indri is flexible enough to handle other parsers easily. Indri includes stopword removal and stemming tools as well.

The indexer stores the index in memory as much as possible in order to keep disk I/O to a minimum. All disk writes happen in a background thread, so queries and indexing do not have to wait on I/O operations as long as memory is available.

The query system parses the Indri query language into an intermediate representation, which is then optimized. It also understands which operators will need more than one query pass when used in a cluster, and those operations are performed transparently.

Indri can be used directly from C++. All the Indri API calls are available from Java as well, and all query capabilities are available from PHP. This allows Indri to integrate well into most workflows.

4. Conclusion

Recent experiments at the TREC conference (Metzler et al. 2004b) indicate that Indri is capable of best-in-class retrieval effectiveness, while still being competitive in retrieval efficiency. Indri is still a new system, so we expect it to improve rapidly in the future.

Inquery started a tradition at the University of Massachusetts of building information retrieval systems that are simultaneously useful for academic, intelligence and corporate tasks. Indri continues that tradition by incorporating new features that address new research areas in information retrieval.

Acknowledgements

This work was supported in part by the Center for Intelligent Information Retrieval and in part by Advanced Research and Development Activity 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

J. P. Callan, W. B. Croft, and J. Broglio. TREC and TIPSTER Experiments with INQUERY. In *Readings in Information Retrieval*, ed. Karen Sparck Jones and Peter Willett, 436-445. San Francisco, CA: Morgan Kaufmann, 1997. [Originally published in: *Information Processing & Management* 31 (1995): 327-332.

D. Harman, The DARPA TIPSTER Project, SIGIR Forum, 26, 26-28 (1992).

D. Metzler, V. Lavrenko and W.B. Croft. Formal multiple Bernoulli models for language modeling. In *SIGIR 2004*, pp. 540-541.

D. Metzler, T. Strohman, H. Turtle, and W. B. Croft. “Indri at TREC 2004: Terabyte Track,” to appear in the Online Proceedings of 2004 Text REtrieval Conference (TREC 2004).

J. Ponte and W. B. Croft, A language modeling approach to information retrieval. In *SIGIR 1998*, pp. 275-281.

H. Turtle and W. B. Croft, Evaluation of an inference network based retrieval model. *Trans. Inf. Syst.*, 9(3):187-222, 1991.