Towards Practical Relevance Ranking for 10 Million Books

www.hathitrust.org

Tom Burton-West
Information Retrieval Programmer
Digital Library Production Service
University of Michigan Library
www.hathitrust.org/blogs/large-scale-search

Code4lib
February 12, 2013
HathiTrust

- HathiTrust is a shared digital repository
- 70+ member libraries
- Large Scale Search is one of many services built on top of the repository
- Currently about 10.5 million books
- 450 Terabytes
  - Preservation page images; jpeg 2000, tiff (438TB)
  - OCR and Metadata about (12TB)
Large Scale Search Challenges

• **Goal:** Design a system for full-text search that will scale to 10 million -20 million volumes (at a reasonable cost.)

• **Challenges:**
  – Multilingual collection (400+ languages)
  – OCR quality varies
  – Very long documents compared to IR research collections and most large-scale search applications
  – Books are different!
Relevance Ranking Questions

• How should MARC metadata fields be scored relative the full-text OCR?
• How should we tune relevance ranking to properly accommodate book length documents?
• If we break books down into smaller parts (chapters, sections, pages), how should the relevance scores for the parts be combined to rank the books?
• How do we test any of the above in a principled way?
Relevance Ranking for Books

- HT average document size huge compared to IR research collections.
- Solr’s default algorithm ranks very short documents much too high.
- 2007 IBM TREC results: Modifications to Lucene’s default length normalization resulted in relevance ranking comparable to state-of-the-art.
- Solr 4 implements a number of modern ranking algorithms which have parameters to allow tuning for document length characteristics.
Long Documents

- Average HathiTrust document is 760KB containing over 100,000 words.
  - Estimated size of 10 million Document collection is 7 TB.
- Average HathiTrust document is about 30 times larger than the average document size of 25KB used in Large Research test collections
  - Over 100 times larger than TREC ad hoc

<table>
<thead>
<tr>
<th>Collection</th>
<th>Size</th>
<th>Documents</th>
<th>Average Doc size</th>
</tr>
</thead>
<tbody>
<tr>
<td>HathiTrust</td>
<td>7 TB</td>
<td>10 million</td>
<td>760 KB</td>
</tr>
<tr>
<td>ClueWeb09 (B)</td>
<td>1.2 TB</td>
<td>50 million</td>
<td>25 KB</td>
</tr>
<tr>
<td>TREC GOV2</td>
<td>0.456 TB</td>
<td>25 million</td>
<td>18 KB</td>
</tr>
<tr>
<td>TREC ad hoc</td>
<td>0.002 TB</td>
<td>0.75 million</td>
<td>3 KB</td>
</tr>
<tr>
<td>HathiTrust (pages)</td>
<td>7 TB</td>
<td>3,700 million</td>
<td>2KB</td>
</tr>
</tbody>
</table>
TF*IDF ranking

• Solr/Lucene’s relevance ranking formula is loosely based on the vector space model, which is one of the tf*idf families of ranking algorithms.

• TF = term frequency.
  – The more often a query term occurs in a document the more likely that the document is relevant.

• IDF = inverse document frequency.
  – The fewer documents that contain a query term, the better the term is for discriminating between relevant and irrelevant documents.

• Length normalization
  – Adjusts scores to account for different document lengths.
Solr’s aggressive Length normalization makes short documents rank too high

• Search for the word “book” in HathiTrust
• Highest ranked document contains just 4 words of OCR “The Book of Job”
• Search for word “Dog”
• 3 of top 5 documents contain less than 1,500 words. (Average doc contains 100,000)
Preliminary tests with Solr 4

• Indexed 1 shard of data (850,000 docs) with 3 new algorithms (using default parameters)
  – BM25, DFR, IB
  – Compared with same data indexed with Solr/Lucene default algorithm

• Preliminary tests
  – Ran a few queries and looked at top 10 results
  – None of these algorithms had the same problem as the default Lucene/Solr algorithm with very short documents
  – No other *obvious* difference in quality of results
  – Need more systematic testing!
Parameter Tuning

• Modern algorithms in Solr 4 have parameters to tune TF normalization and length normalization
• Defaults based on training with short TREC documents (average 300-1600 words) unlikely to work for 100,000 word books
• Need a training/test collection of books
Complications: Dirty OCR

• Dirty OCR can distort document statistics used in ranking
  – Taghva et. al. found that images misrecognized as text could increase the number of words in a document by 30%.
  – MaxTF or averageTF for a document can also be affected by dirty OCR
Complications: Multiple Languages

- Indexing all 400 languages in one index distorts IDF statistics
  - A query for [die hard] will use an IDF for “die” that includes the number of documents containing the German word “die”
  - A query for the Swedish word for ice “is” will use an IDF that includes the counts for documents containing the English word “is”.
Books are Different:
TF in Chapters vs Whole Book

Montemuro and Zanette (2009)
Books are Different: Should we index parts of books?

- What unit should we use for indexing?
  - Whole book, chapters, sections, pages, other units?
- Do user’s want a ranked list of books, or chapters or pages or snippets?
  - Depends on user need and context.
    - factual questions “Capital of Canada”
    - big questions: “causes of the English civil war”, “relationship of fat in diet to serum cholesterol to heart disease.”
  - Depends on type of document
    - Bound journals (2.5 million in HT) should be indexed by article
    - Dictionaries and encyclopedias (over 100,000 in HT) should be indexed per entry
    - Reference books should also be indexed per entry (Bates 1986)
Should we index parts of books?
Practical issues

• Chapter markup is based on OCR and likely unreliable. If 10% of volumes incorrectly partitioned, they would not be ranked correctly
• Structural metadata based on OCR. Journal article boundaries, or encyclopedia entries, not marked up in metadata.
• Instead of book chapters could try to segment by “Topic.”
Should we index parts of books?
Practical issues

• We have good mark-up for whole volumes and pages, so we could index pages.
• Will Solr scale to 3.7 Billion pages (with current hardware?)
• Until recently Solr did not support part-whole relationships. “Field-Collapsing” could be used to group pages into books. Will it scale?
INEX Book Track

- Collection of 50,000 books with OCR and MARC data used for main book retrieval task 2007-2010
- Ongoing issues with low active participation rates and insufficient relevance judgments
Questions investigated by the INEX Book Track participants

- What is best unit to use in indexing?
  - Whole book, groups of pages, pages?
  - Considered chapters but no one used them!
  - Is best unit affected by query length?

- What is the best way to combine page scores to rank books?
  - Ranking by highest ranking page in book not often the best

- How to best to use OCR and MARC metadata in scoring.

Results contradictory and inconclusive

- Could not tune algorithms for document length and collection characteristics without training corpus with judgments.

- Several groups used ranking algorithms with defaults which were based on 300-1000 word TREC documents, not 100,000 word books.

- Not enough relevance judgments!
Tuning Relevance Ranking

Current Method: Ad hoc relevance testing

- Set some boost values
- try out some queries
- repeat until results look good
- Ask for user/librarian testing comments
- Much of the testing based on known item queries.
Relevance Testing Plan

• Create a representative set of queries
• Improve live monitoring and testing
  – Query log metrics (click logs)
  – Framework for A/B testing with interleaving
• Create a test collection
Relevance Testing: Queries

• We need a collection of test queries that reflect different types of user needs
  – Query log analysis
  – User studies
  – We can use the test queries for both more systematic ad hoc testing and as a basis for a test collection.

• We will add click logging to our search logs
  – Allows some measure of how well our ranking is working
    • Click on top 3 hits
    • Various click/relevance models
Testing Relevance

• Online Evaluation and A/B testing
  – can only test two different algorithms at a time
  – risky if doing live testing
  – good for fine tuning but not for parameter sweep
• Offline testing (Test collection)
  – Set of queries, Set of documents, Set of relevance judgments
  – Re-usable
  – Can test many algorithms with many parameter variations in batch mode
Test Collection

• Queries
  – Need sufficient number of representative queries
    • 50 -100 is probably the minimum.
  – Queries must address range of use cases/user needs

• Collection of Documents
  – Need representative collection that is small enough to work with, but large enough to infer that results will apply to entire 10 million document collection
Test Collection
Relevance Judgments

• Collecting Relevance judgments is labor intensive
  – TREC hires 8-10 retired intelligence analysts to do the judging
  – Sanderson (2010) estimated 75 person days for a typical 50 topic TREC track. (This is for short documents)
  – Kazai reports significantly more effort required for relevance judgments of books
  – Google and Bing hire many workers to make judgments
Test Collection volunteers needed

• If you are interested in helping to organize gathering relevance judgments from librarians and users, please contact me

tburtonw@umich.edu
Thank You!

Tom Burton-West
tburtonw@umich.edu

www.hathitrust.org/blogs/large-scale-search
References

• INEX book track: https://inex.mmci.uni-saarland.de/data/publications.jsp
• Grant Ingersoll on Relevance testing: http://searchhub.org/2009/09/02/debugging-search-application-relevance-issues/
References

- **Solr/Lucene’s default ranking algorithm**
  - [http://lucene.apache.org/core/4_1_0/core/org/apache/lucene/search/similari+es/TFIDFSimilarity.html](http://lucene.apache.org/core/4_1_0/core/org/apache/lucene/search/similari+es/TFIDFSimilarity.html)
- **New ranking algorithms in Solr/Lucene 4.x**
  - [http://lucene.apache.org/core/4_1_0/core/org/apache/lucene/search/similari+es/package-summary.html#package_description](http://lucene.apache.org/core/4_1_0/core/org/apache/lucene/search/similari+es/package-summary.html#package_description)
- **Solr field collapsing**
  - [http://www.searchworkings.org/blog/-/blogs/24078](http://www.searchworkings.org/blog/-/blogs/24078)
- **Length Normalization**
References: Relevance Testing

- **Click logs and other online evaluation techniques**

- **Side by side evaluation**

- **User studies**
References: Test Collections


• Ellen Voorhees and Donna Harman, editors. TREC: Experiment and Evaluation in Information Retrieval. The MIT Press, 2005


• Google and Bing
  – http://searchengineland.com/interview-google-search-quality-rater-108702
  – http://www.youtube.com/watch?v=nmo3z8pHX1E

• Crowdsourcing relevance judgments
  – Gabriella Kazai’s publication list: http://www.gabriella-kazai.com/