Pruning Long Documents for Distributed Information Retrieval

Jie Lu
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
jielu@cs.cmu.edu

Jamie Callan
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
callan@cs.cmu.edu

ABSTRACT

Query-based sampling is a method of discovering the contents of a text database by submitting queries to a search engine and observing the documents returned. In prior research sampled documents were used to build resource descriptions for automatic database selection, and to build a centralized sample database for query expansion and result merging. An unstated assumption was that the associated storage costs were acceptable.

When sampled documents are long, storage costs can be large. This paper investigates methods of pruning long documents to reduce storage costs. The experimental results demonstrate that building resource descriptions and centralized sample databases from the pruned contents of sampled documents can reduce storage costs by 54-93% while causing only minor losses in the accuracy of distributed information retrieval.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Performance, Experimentation

Keywords

Distributed Information Retrieval, Document Pruning

1. INTRODUCTION

Distributed IR has emerged as a way of providing ad-hoc search capabilities in environments that contain many searchable text databases ("search engines") [2, 3, 6, 12]. The goal is to provide a single interface for ad-hoc, multi-database search. A central site (a "distributed search engine") selects an appropriate set of databases for each query ("resource selection"), submits the query to the selected databases, and merges the ranked lists of documents returned by each database into a single, integrated ranked list ("result merging").

Resource selection requires *resource descriptions* that describe the contents of each database. In a cooperative environment (e.g., an intranet), each database provides an accurate resource description when requested. In an uncooperative environment (e.g., the Internet), the distributed search engine must discover the contents of each database, for example, by submitting queries and observing what is returned ("query-based sampling"). The two approaches produce similar results for most queries [3, 4].

Our research interest is distributed IR in large-scale, uncooperative environments, so we have worked extensively with query-based sampling of databases. Because the only information needed to construct the resource descriptions used by most resource selection algorithms is term and term frequency information, it is not necessary to store the original (sampled) documents after they are analyzed. However, some solutions to other distributed IR tasks, for example query expansion [10] and result merging [9], require more information. Our solution was to store the sampled documents in a centralized sample database. A *centralized sample database* is a corpus created by sampling documents from all of the databases available. For some tasks it is a useful surrogate for the whole distributed IR environment.

The centralized sample database must satisfy two requirements: i) it must represent each individual database relatively accurately, and ii) the total centralized sample database size must not be too large, for efficiency and scalability reasons. If the environment contains 1,000 databases, and if 300 documents are sampled from each database (a typical sample size in our research), then the centralized sample database contains 1,000 $\times$ 300 = 300,000 documents. This is not large by IR standards.

The use of query-based sampling to build resource descriptions and centralized sample databases works well for TREC data [9, 10]. The real world presents different challenges. The U.S. Government Printing Office (USGPO) provides access to almost 2,400 databases of government documents and publications. Many of these documents are much longer than TREC documents. For example, in one of our tests with USGPO data, the typical sampled document size was 440 kilobytes (KB) on average. The estimated size of a centralized sample database for this environment is 2,400 databases $\times$ 300 documents $\times$ 440 KB = 302 gigabytes. We would prefer it not to use so much space.

Resource descriptions created by query-based sampling are *partial* resource descriptions, because they are based on only a small subset of the documents in a database. This paper studies the effects of further reducing the sizes of resource descriptions and centralized sample databases by pruning terms from the documents that are used to build them. In particular, we show that pruned representations allow a large decrease in the sizes of these data resources while causing only a small reduction in resource selection accuracy and overall retrieval performance.

The following section describes related work. Section 3 outlines different methods of pruning documents. Sections 4 and 5 discuss the data resources and evaluation methodologies used in our experiments. Experimental results are presented in Section 6. Section 7 concludes.

2. RELATED WORK

An IR system's storage space can be reduced by reducing index sizes. Reducing index sizes is an old topic in IR, encompassing techniques as varied as compression, term removal (e.g., stopword lists), and pruned posting lists (i.e., recording some occurrences of a term but not others). The latter set of techniques is the most relevant to the research reported in this paper. A recent and illustrative example of pruning postings is described in [5]. Given a cutoff threshold, if the score of a term in a document is smaller than the threshold, the posting is eliminated from the index. The posting list of each term is pruned independently of other terms. Index size reductions of 60-70% degraded average precision about 10% for short queries.

One can view the pruning of posting lists as generating document summaries by removing terms from documents. There is a large body of prior research on document summarization, and some prior research suggests that ad-hoc retrieval from indexes of document summaries is as effective as retrieval from indexes of complete document texts [11]. In general index pruning and document summarization have been viewed as separate problems with different solutions, because index pruning may produce a summary that is useful for some task, but unintelligible. However, the prior utility of human-readable summaries for ad-hoc retrieval [11] and query expansion [8] suggests that they can also be considered for reducing index sizes.

Much of the recent summarization research has been on techniques that select sentences from the document to generate either *query-biased* or *generic* summaries. We are particularly interested in using Luhn's keyword-based clustering measure, which was shown recently to produce summaries that support effective query-expansion [8]. We describe this measure in Section 3.3. It is an open question whether such summaries would also be useful for resource selection, or for use in a centralized sample database that supports result merging.

Pruning terms from documents generates partial document representations. Craswell et al [6] showed that result merging in a distributed IR environment can be performed with only partial document representations. In their distributed IR experiments, databases were selected, searched, and then either the complete document or just the first part of the document was downloaded. A tf.idf score was calculated for the downloaded texts, and this "global" document score was used for merging results from different databases. The use of partial document representations degraded accuracy by less than 10%. This approach of result merging is somewhat costly in terms of communication and computation, but it demonstrates that partial document representations can be effective for distributed IR tasks.

All in all, the prior research suggests that i) index sizes can be reduced (without hurting much accuracy) by using partial document representations, and ii) document summarization techniques may be effective on creating partial document representations that are effective for distributed IR tasks.

3. PRUNING METHODS

Our goal is to reduce the sizes of long documents without reducing their utility for building resource descriptions and centralized sample databases that support other retrieval tasks. We call this *pruning* by removing words that are not selected.

The pruning methods presented here fall into two categories: frequency-based and location-based. *Frequency-based* pruning uses term frequency information to measure the importance of the terms and to prune relatively unimportant terms or segments of the document. Frequency-based methods favor concepts that occur repeatedly in a document, and so might be expected to better represent its contents as a whole. *Location-based* pruning selects terms based on their positions in the document. Location-based pruning is simple and fast, and typically favors the beginning (or other specified) parts of a document.

The pruning methods can also be distinguished based on whether multiple occurrences of a term are allowed in the pruned document (*multiple-occurrences* methods vs. *single-occurrence* methods). A multiple-occurrences method will better reflect term frequency distributions, but may select fewer unique terms than a single-occurrence method.

Below we present eight methods of pruning documents. Four are frequency-based (TF, TFIDF, LUHNM, and LUHNS) and four are location-based (FIRSTM, FIRSTS, RANDM and RANDS). Three are multiple-occurrences methods (LUHNM, FIRSTM and RANDM) and five are single-occurrence methods (TF, TFIDF, LUHNS, FIRSTS and RANDS). All eight content pruning methods prune each document independently of other documents.

3.1 Term Frequency (TF)

Given a threshold *r*, the *term frequency* pruning method ranks all unique non-stopword terms in the document by within-document term frequencies and selects terms ranked higher than *r*. Each term occurs only once in the pruned document. The method is based on the idea that within-document frequency is an indicator of the importance of a non-stopword term in the document.

3.2 Term Frequency and Inverse Document Frequency (TFIDF)

The motivation of the *tf.idf* pruning method came from the widely used tf.idf term weighting scheme in retrieval tasks. It ranks all unique non-stopword terms in the document by their tf.idf scores calculated as:

$$S = (\log tf + 1) \times (\log \frac{N}{df} + 1)$$

where *tf* is the within-document term frequency, *df* is the number of sampled documents that contain the term, and *N* is the number of documents sampled from the database.

Given a ranking threshold *r*, terms ranked higher than *r* are selected and the rest are discarded. Each term occurs only once in the pruned document.

3.3 Luhn's Keyword-Based Clustering (LUHNM and LUHNS)

Luhn's keyword-based clustering measures the importance of a sentence in the document [8]. It is used, for example, to select sentences for use in a document summary [8]. Other summarization techniques could also be used. In this paper, we

Table 4.1 Summary statistics for the databases in Testbed I.

Type of docs	# dbs	# docs		Avg doc len (words)	
		min	max	min	max
short	8	36,946	39,713	24	129
medium	72	8,039	13,602	399	603
long	13	2,999	4,124	1,210	1,701
very long	7	754	1,236	4,011	6,408

are interested in the effectiveness of using summaries for distributed IR tasks. The comparison of different summarization techniques in this environment is not our research focus here.

A sentence has a high score if it has a dense cluster of "significant" words. A word is classified as "significant" if it is not a stopword and its within-document term frequency is larger than a threshold T defined as $T = 7 + I \times 0.1 \times | L - n |$ where n is the number of sentences in the document, L is 25 for $n < 25$ and 40 for $n > 40$, $| L - n |$ is the absolute value of $L - n$, and I is 0 for $25 \le n \le 40$ and 1 otherwise.

Clusters of significant words are created such that significant words are separated by no more than 5 insignificant words within the sentence. The significance score for a cluster is $S = SW^2 / TW$ where SW is the number of significant words in the cluster (both cluster boundaries are significant words), and TW is total number of words in the cluster.

The sentence score is defined as the score of the most significant cluster it contains. Sentences are ranked by their scores. Given a threshold number of terms t, if multiple occurrences of a unique term are allowed, non-stopword terms are selected from top-ranked sentences until the number reaches t (LUHNM); otherwise, unique non-stopword terms from top-ranked sentences are selected until the number reaches t (LUHNS).

3.4 First Part of the Document (FIRSTM and FIRSTS)

The *first part of the document* pruning methods select the first t non-stopword terms if multiple occurrences of a unique term are allowed (FIRSTM), or the first t unique non-stopword terms otherwise (FIRSTS). The rest of the terms are discarded. These methods simulate an environment in which only the first part of a document is downloaded during query-based sampling, for example to reduce communication costs.

3.5 Randomly Selected Terms of the Document (RANDM and RANDS)

The *randomly selected terms* pruning methods choose positions in the document randomly and select all corresponding non-stopword terms if multiple occurrences of a unique term are allowed (RANDM), or those corresponding non-stopword terms that haven't been selected for this document otherwise (RANDS) until the number reaches t. The rest of the terms are discarded. Although the selection criterion is location-based, it can also be regarded as frequency-based because high frequency terms occur in more locations and thus are more likely to be selected.

4. MULTI-DATABASE TESTBEDS

The pruning methods were tested on two distributed IR testbeds. Testbed I consists of 100 databases obtained by dividing data from TREC CDs 1, 2, and 3 based on source and publication date. This testbed has been used in prior research on query-based sampling, resource selection, and result merging in distributed information retrieval [3, 9]. These 100 databases can be grouped based on average document length and total number of documents. Summary statistics are shown in Table 4.1.

TREC topics 51-150 were used to run our experiments. These topics were chosen because they have relevance judgments for all parts of TREC CDs 1, 2, and 3. Only title fields were used in our experiments, so queries are short. The standard TREC relevance assessments supplied by the U. S. National Institute for Standards and Technology were used [7].

Testbed II is a variation of Testbed I created specifically to test the effects of pruning in environment where some databases have extremely long documents (i.e., to simulate an environment like the USGPO). Documents from the *1988 Federal Register* were grouped by content, using k-means clustering algorithm. Documents within a cluster were sorted into chronological order, and then subsequences were concatenated to create very long documents. The resulting *fr88_merge* database had 2,357 documents with an average length of 12,854 terms. The same process was applied to the *1993 U.S. Patents* documents. The resulting *patent_merge* database had 2,777 documents with an average length of 12,047 terms. The *fr88_merge* and *patent_merge* databases replaced 13 corresponding databases in Testbed I, resulting in Testbed II, which contains 89 databases. Summary statistics are shown in Table 4.2.

Table 4.2 Summary statistics for the databases in Testbed II.

Type of docs	# dbs	# docs		Avg doc len (words)	
		min	max	min	max
short	8	36,946	39,713	24	129
medium	72	8,039	13,602	399	603
long	7	3,547	4,124	1,353	1,618
very long	2	2,357	2,777	12,047	12,854

The relevance assessments for Testbed II were adjusted accordingly. If any component of a very long "concatenated" document was relevant to a topic, the very long document was also considered relevant. This is consistent with NIST assessment methodology in which a document is judged relevant if any part of it is relevant [7].

5. EVALUATION METHODOLOGY

Distributed IR systems are affected by the performance of each individual component as well as the interaction between different components. Past research showed that it is important to evaluate directly where a change is made in a system, and also to evaluate the effects of the change on "downstream" components [3]. In our case, that means evaluating how well pruned resource descriptions match unpruned resource descriptions, and also how they affect the accuracies of resource selection and document retrieval. Each type of evaluation is discussed below.

5.1 Measuring the Accuracy of Learned Resource Descriptions

Resource descriptions are usually based on vocabulary and term frequency information [3, 12]. Prior research on query-based sampling [3] compared resource descriptions using *ctf ratio* and the *Spearman rank correlation coefficient*. Adopting the terminology of that research, we call these the "*actual*" description (created from all documents of a database) and the "*learned*" description (created from sampled documents).

The *ctf ratio* (collection term frequency ratio) measures how well the learned vocabulary matches the actual vocabulary. It is the proportion of term occurrences in the database that are covered by terms in the learned resource description. A ctf ratio of 80% means that the learned resource description contains the terms that account for 80% of all term occurrences in the database. For a learned vocabulary *V'* and an actual vocabulary *V*, ctf ratio is:

$$\frac{\sum_{i \in V'} ctf_i}{\sum_{i \in V} ctf_i}$$

where ctf_i is the collection term frequency of (unique) term *i*.

The *Spearman rank correlation coefficient* compares the orderings of terms ranked by frequency in learned resource description and actual resource description. It is defined as:

$$\frac{1 - \frac{6}{n^3 - n}\left(\sum_i d_i^2 + \frac{1}{12}\sum_k (f_k^3 - f_k) + \frac{1}{12}\sum_m (g_m^3 - g_m)\right)}{\sqrt{1 - \frac{\sum_k (f_k^3 - f_k)}{n^3 - n}} \times \sqrt{1 - \frac{\sum_m (g_m^3 - g_m)}{n^3 - n}}}$$

where:

- d_i is the rank difference of common (unique) term *i;*
- *n* is the number of unique terms in the learned resource description;
- f_k is the number of ties in the k^{th} group of ties in the learned resource description; and
- g_m is the number of ties in the m^{th} group of ties in the actual resource description.

The rank correlation coefficient has value 1 when two orderings are identical, –1 when they are in reverse order, and 0 when they are uncorrelated.

Stopwords were removed from both the learned and actual language models prior to comparison with ctf ratio and Spearman rank correlation coefficient.

5.2 Measuring the Accuracy of Database Rankings

The database ranking metric R_n is a standard metric in distributed IR research [3]. It is analogous to the *Recall* metric for document rankings. If the desired database ranking is a relevance-based ranking that ranks databases by the number of relevant documents they contain for a query, R_n is defined as:

$$R_n = \frac{\sum_{i=1}^{n} \text{NumRel}(db_{e_i})}{\sum_{i=1}^{n} \text{NumRel}(db_{r_i})}$$

where:

- *n* is the number of databases to be ranked;
- db_{e_i} is the i^{th} ranked database in estimated database ranking;
- db_{r_i} is the i^{th} ranked database in relevance-based ranking.

5.3 Measuring the Accuracy of Document Rankings

The quality of document rankings is measured by the well-known *Precision* and *Recall* metrics. We measured precision at document ranks 5, 10, 15, 20, 30, and 100.

6. EXPERIMENTAL RESULTS

A series of experiments was conducted to study the effects of the pruning methods described in Section 3. Resource descriptions and a centralized sample database were created using (pruned or unpruned) documents obtained by query-based sampling [4]. Query-based sampling was conducted with single-term queries selected randomly from documents already sampled from the database, and 4 documents examined per query. 300 documents were sampled from each database. This methodology is consistent with prior research [3, 9, 10].

For Testbed II, sampled documents were pruned only if they belong to the "very long documents" databases *fr88_merge* and *patent_merge*. This choice guaranteed that any differences would be due only to pruning very long documents. The average length of documents sampled from Testbed I was around 1,600 terms (about 400 unique terms), so the pruning thresholds for the "very long documents" databases in Testbed II were set to 1,600 for multiple-occurrences methods and 400 for single-occurrence methods. The average length of documents sampled from "very long documents" databases *fr88_merge* and *patent_merge* was 13,292 terms, so the pruning ratios were about 8:1 for multiple-occurrences methods and 33:1 for single-occurrence methods.

For Testbed I, all sampled documents were subject to pruning. The pruning thresholds for Testbed I were chosen to yield the same pruning ratios on "long documents" and "very long documents" databases as were used in Testbed II. The average length of documents sampled from "long documents" and "very long documents" databases in Testbed I was 3,332 terms. Using the (approximately) 8:1 pruning ratio (multiple-occurrences methods) and 33:1 pruning ratio (single-occurrence methods) results in thresholds of 400 terms for multiple-occurrences methods and 100 terms otherwise.

Databases were ranked with the well-known CORI database ranking algorithm. CORI has been described often in the research literature [2, 3, 9, 10], so we only sketch it briefly. The belief of database db_i with respect to query term r_k is $p(r_k \mid db_i)$ defined as:

$$p(r_k \mid db_i) = b + (1 - b) \times T \times I$$

$$T = \frac{df_i}{df_i + 50 + 150 \times cw_i / avg\_cw}$$

$$I = \frac{\log\left(\frac{|DB| + 0.5}{cf}\right)}{\log(|DB| + 1.0)}$$

where:

- df_i is the number of documents in db_i that contain r_k;
- *cf* is the number of databases that contain r_k;
- $|DB|$ is the number of databases to be ranked;
- cw_i is the number of words in db_i;
- *avg_cw* the average *cw* of the databases to be ranked;
- *b* is the default belief, usually 0.4.

The combination of belief values for query terms is the average of their beliefs in our experiments.

The 10 top-ranked databases for a query were selected for search, as in prior distributed IR research [2, 3, 9, 10]. The selected databases were searched using the INQUERY search engine [1].

Document rankings produced by different databases must be merged into a single ranking. The CORI result-merging algorithm is a standard heuristic used with the CORI database ranking and

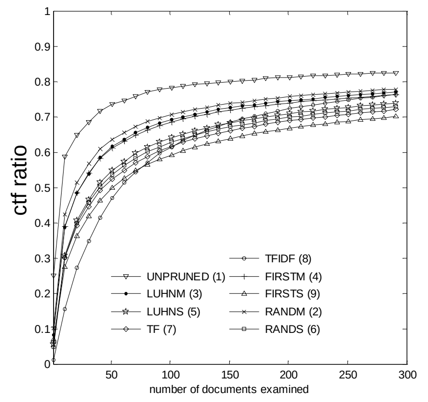

Figure 6.1a Average ctf ratio of resource descriptions learned for 13 "long documents" databases in Testbed I.

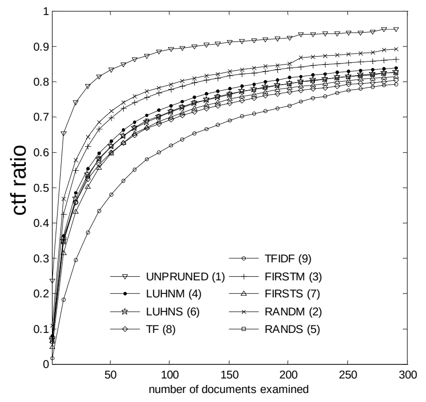

Figure 6.1b Average ctf ratio of resource descriptions learned for 7 "very long documents" databases in Testbed I.

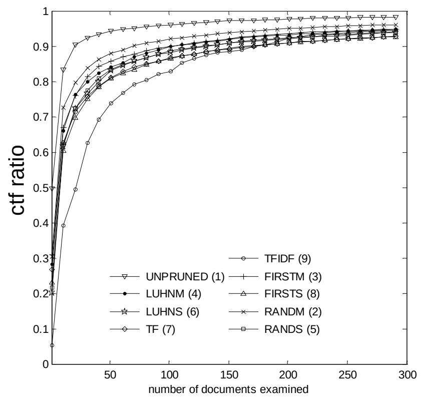

Figure 6.1c ctf ratio of resource descriptions learned for fr88_merge in Testbed II.

INQUERY document retrieval algorithms [2, 3, 9, 10], so we used it in the research reported here. A "normalized" document score *D'* suitable for merging is calculated as [3]:

$$D' = \frac{\frac{D - D_{\min}}{D_{\max} - D_{\min}} \times (1 + 0.4 \times \frac{R_i - R_{\min}}{R_{\max} - R_{\min}})}{1.4}$$

where:

D is the document score returned by database *i*;

R_i is the ranking of database *i*;

$R_{\max}$ and $R_{\min}$ are the highest and lowest scores respectively that the database ranking algorithm could potentially assign to a database;

$D_{\max}$ and $D_{\min}$ are the highest and lowest document scores respectively that are returned by database *i*.

We also used a newer result-merging algorithm that on a query-by-query basis uses a centralized sample database to generate unsupervised training data for a regression-based normalizing function [9]. Sampled documents from all databases are combined into a single centralized sample database, and this database is indexed with INQUERY. After 1,000 documents were retrieved from a selected database, another 5,000 documents were retrieved from the centralized sample database. Documents that occurred in both lists ("overlap documents") served as training data for a linear regression algorithm that learned to normalize document scores returned by this database (see [9] for details). Each selected database learned its own normalizing function. Finally a merged ranking of 100 documents were returned for each query based on the normalized document scores.

The regression-based result-merging algorithm was chosen because the centralized sample database introduces another potential sensitivity to pruned documents. Retrieval of pruned documents might generate less accurate document rankings, which would reduce the effectiveness of result merging.

6.1 Effects of Content Pruning on the Accuracy of Resource Descriptions

Our first set of experiments measured how well resource descriptions created from pruned documents matched resource descriptions created from unpruned documents. In this set of experiments we studied only "long documents" and "very long documents" databases (Section 4), because very few documents in "short documents" and "medium documents" databases were long enough to be pruned. The experimental results for both testbeds are summarized in Figures 6.1 (ctf ratio) and 6.2 (Spearman rank correlation coefficient). The baselines were generated from unpruned documents. In some cases it is difficult to tell curves apart, so each curve's rank at the "100 sampled documents" point is shown in parenthesis. Because results for *fr88_merge* and *patent_merge* in Testbed II are very similar, only results for *fr88_merge* are shown here due to limited space.

The *ctf ratio* measures how well a resource description represents the vocabulary of a database. The results for the ctf ratio show that even using pruned documents, the vocabulary that represents at least 70% of the non-stopword term occurrences in each database can still be identified quickly. After sampling 300 documents, the difference between ctf ratios for resource descriptions produced from pruned documents and those produced from unpruned documents is less than 15%.

Methods based on randomly selected terms are better than those based on Luhn's keyword-based clustering and the first part of the document. The difference between multiple-occurrences methods was usually small. The same is true for single-occurrence methods. The difference between multiple-occurrences and single-occurrence methods is due to the larger vocabulary found using multiple-occurrences methods with given thresholds. On Testbed I the average number of unique terms in pruned documents using multiple-occurrences methods with threshold 400 is 193 compared with 100 using single-occurrence methods. On Testbed II, the average number of unique terms of pruned documents using multiple-occurrences methods with threshold 1,600 is 530 compared with 400 using single-occurrence methods.

The *Spearman rank correlation coefficient* measures how well a resource description represents the term frequency distribution in a database. Accumulating more terms during sampling may not necessarily lead to more accurate orderings of terms by frequency in learned resource description. This explains why the curve of the Spearman rank correlation coefficient is not always monotonic. The result that sometimes the unpruned performed a little bit worse than the pruned is due to the same reason.

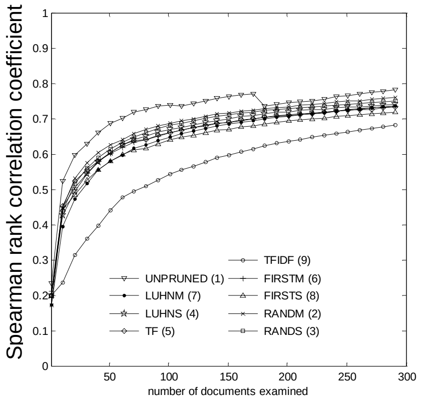

Figure 6.2a Average Spearman rank correlation coefficient of resource descriptions learned for 13 "long documents" databases in Testbed I.

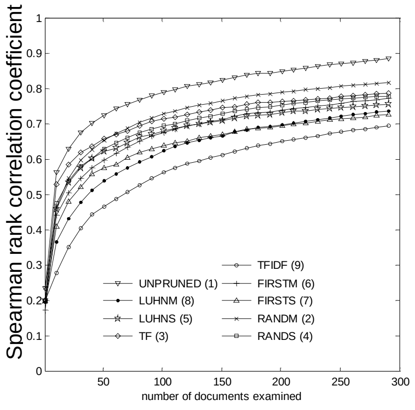

Figure 6.2b Average Spearman rank correlation coefficient of resource descriptions learned for 7 "very long documents" databases in Testbed I.

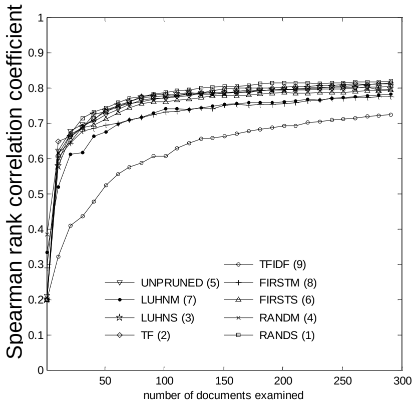

Figure 6.2c Spearman rank correlation coefficient of resource description learned for fr88_merge in Testbed II.

Methods based on randomly selected terms appear to generate more accurate orderings of terms by frequency than those based on Luhn's keyword-based clustering and the first part of the document. However, unlike with *ctf ratio*, it is not necessarily the case that multiple-occurrences methods worked better than single-occurrence methods. Fairly accurate estimation of term orderings by frequency at the database level can still be obtained by single-occurrence methods (except TFIDF) even if we ignore within-document term frequency when pruning documents.

6.2 Effects of Content Pruning on the Accuracy of Database Rankings

A second set of experiments studied the effects of the pruned resource descriptions on the ability of the CORI algorithm to rank databases. Figure 6.3 summarizes the recall values of database rankings for Testbeds I and II. The rank of each curve at the "10 selected databases" point is provided in parenthesis.

For Testbed I, multiple-occurrences methods had higher *Recall* than single-occurrence methods, presumably due to better term frequency information. The difference between using pruned and unpruned contents for resource descriptions was less than 10%. For Testbed II, there was hardly any difference in recall using different pruning methods. The difference between using pruned and unpruned contents was also rarely observable. If we focus on the top 10 of the ranking, still multiple-occurrences methods had higher recall than single-occurrence methods. In general, content pruning didn't result in much degradation of database rankings.

6.3 Effects of Content Pruning on the Accuracy of Document Rankings

A third set of experiments studied the effects of the pruned resource descriptions on how well two result-merging algorithms merge document rankings. The experimental results are summarized in Tables 6.1 (Testbed I) and 6.2 (Testbed II). The baseline results were generated from unpruned documents.

In general, the effect of pruning on the accuracy of document rankings was consistent with its effect on the accuracy of database rankings. The results indicate that creating resource descriptions from pruned documents led to less than 10% (usually less than 5%, which few users would notice) relative performance loss using the LUHNM, LUHNS, RANDM, and FIRSTM pruning methods. The performance loss using RANDS, FIRSTS, TF, and TFIDF methods was usually less than 15%.

Generally, multiple-occurrences methods provided better accuracy (less degradation) than single-occurrence methods. However, the

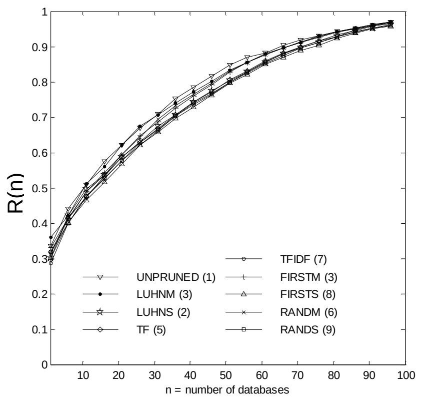

Figure 6.3a Database ranking recall for Testbed I.

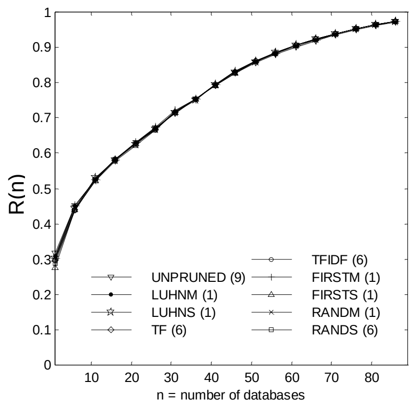

Figure 6.3b Database ranking recall for Testbed II.

Table 6.1a Precision of document rankings created by the CORI result-merging algorithm on Testbed I.

Rank	Full-	Precision							
	content	TF	TFIDF	LUHNM	LUHNS	FIRSTM	FIRSTS	RANDM	RANDS
5 docs	0.438	0.390 (-10.9%)	0.396 (-9.6%)	0.414 (-5.5%)	0.428 (-2.3%)	0.414 (-5.5%)	0.400 (-8.7%)	0.418 (-4.6%)	0.400 (-8.7%)
10 docs	0.404	0.346 (-14.3%)	0.366 (-9.4%)	0.379 (-6.2%)	0.378 (-6.4%)	0.378 (-6.4%)	0.359 (-11.1%)	0.382 (-5.4%)	0.354 (-12.4%)
15 docs	0.379	0.330 (-12.8%)	0.343 (-9.3%)	0.363 (-4.2%)	0.366 (-3.3%)	0.363 (-4.0%)	0.352 (-7.0%)	0.361 (-4.5%)	0.343 (-9.3%)
20 docs	0.361	0.313 (-13.3%)	0.333 (-7.6%)	0.349 (-3.5%)	0.349 (-3.5%)	0.340 (-5.9%)	0.338 (-6.5%)	0.338 (-6.4%)	0.326 (-9.8%)
30 docs	0.337	0.295 (-12.6%)	0.308 (-8.5%)	0.326 (-3.2%)	0.327 (-3.0%)	0.315 (-6.4%)	0.311 (-7.6%)	0.317 (-5.8%)	0.303 (-10.1%)
100 docs	0.260	0.223 (-14.5%)	0.237 (-8.8%)	0.250 (-4.0%)	0.239 (-8.3%)	0.245 (-5.9%)	0.236 (-9.4%)	0.247 (-5.3%)	0.236 (-9.6%)

Table 6.1b Precision of document rankings created by the regression-based result-merging algorithm on Testbed I.

Rank	Full-	Precision							
	content	TF	TFIDF	LUHNM	LUHNS	FIRSTM	FIRSTS	RANDM	RANDS
5 docs	0.434	0.412 (-5.1%)	0.404 (-6.9%)	0.410 (-5.5%)	0.422 (-2.8%)	0.404 (-6.9%)	0.412 (-5.1%)	0.428 (-1.4%)	0.404 (-6.9%)
10 docs	0.400	0.382 (-4.5%)	0.369 (-7.8%)	0.392 (-2.0%)	0.384 (-4.0%)	0.379 (-5.2%)	0.378 (-5.5%)	0.393 (-1.7%)	0.374 (-6.5%)
15 docs	0.381	0.357 (-6.3%)	0.353 (-7.5%)	0.367 (-3.8%)	0.367 (-3.8%)	0.373 (-2.3%)	0.360 (-5.6%)	0.377 (-1.2%)	0.359 (-5.9%)
20 docs	0.368	0.328 (-10.7%)	0.337 (-8.4%)	0.352 (-4.3%)	0.352 (-4.3%)	0.346 (-5.8%)	0.338 (-8.0%)	0.351 (-4.5%)	0.334 (-9.1%)
30 docs	0.337	0.303 (-10.0%)	0.310 (-7.9%)	0.326 (-3.3%)	0.323 (-4.1%)	0.319 (-5.2%)	0.313 (-7.1%)	0.328 (-2.6%)	0.303 (-10.0%)
100 docs	0.260	0.215 (-17.3%)	0.237 (-8.7%)	0.252 (-3.0%)	0.250 (-3.6%)	0.247 (-4.9%)	0.229 (-11.7%)	0.253 (-2.6%)	0.230 (-14.0%)

Table 6.2a Precision of document rankings created by the CORI result-merging algorithm on Testbed II.

Rank	Full-	Precision							
	content	TF	TFIDF	LUHNM	LUHNS	FIRSTM	FIRSTS	RANDM	RANDS
5 docs	0.436	0.388 (-11.0%)	0.370 (-15.1%)	0.442 (+1.4%)	0.404 (-7.3%)	0.442 (+1.4%)	0.374 (-14.2%)	0.430 (-1.4%)	0.382 (-12.4%)
10 docs	0.411	0.371 (-9.7%)	0.352 (-14.3%)	0.426 (+3.6%)	0.388 (-5.6%)	0.426 (+3.6%)	0.369 (-10.2%)	0.411 (0.0%)	0.371 (-9.7%)
15 docs	0.396	0.355 (-10.3%)	0.341 (-13.8%)	0.405 (+2.2%)	0.367 (-7.2%)	0.405 (+2.2%)	0.353 (-10.8%)	0.396 (0.0%)	0.355 (-10.4%)
20 docs	0.388	0.353 (-9.2%)	0.341 (-12.2%)	0.401 (+3.1%)	0.366 (-5.7%)	0.400 (+2.9%)	0.352 (-9.5%)	0.391 (+0.7%)	0.353 (-9.1%)
30 docs	0.364	0.327 (-10.1%)	0.316 (-13.2%)	0.371 (+1.8%)	0.339 (-6.9%)	0.369 (+1.4%)	0.328 (-10%)	0.361 (-0.9%)	0.331 (-9.1%)
100 docs	0.274	0.249 (-9.0%)	0.246 (-10.0%)	0.277 (+1.2%)	0.251 (-8.3%)	0.277 (+1.2%)	0.248 (-9.5%)	0.270 (-1.3%)	0.249 (-8.9%)

Table 6.2b Precision of document rankings created by the regression-based result-merging algorithm on Testbed II.

Rank	Full-	Precision							
	content	TF	TFIDF	LUHNM	LUHNS	FIRSTM	FIRSTS	RANDM	RANDS
5 docs	0.440	0.428 (-2.7%)	0.430 (-2.3%)	0.440 (0.0%)	0.424 (-3.6%)	0.424 (-3.6%)	0.420 (-4.5%)	0.432 (-1.8%)	0.412 (-6.4%)
10 docs	0.415	0.409 (-1.4%)	0.398 (-4.1%)	0.413 (-0.5%)	0.393 (-5.3%)	0.401 (-3.4%)	0.386 (-7.0%)	0.404 (-2.6%)	0.388 (-6.5%)
15 docs	0.403	0.389 (-3.3%)	0.379 (-5.9%)	0.395 (-2.0%)	0.378 (-6.1%)	0.383 (-5.0%)	0.372 (-7.6%)	0.387 (-4.0%)	0.373 (-7.3%)
20 docs	0.378	0.371 (-1.7%)	0.365 (-3.3%)	0.378 (+0.1%)	0.367 (-2.9%)	0.371 (-1.8%)	0.364 (-3.7%)	0.374 (-1.0%)	0.357 (-5.4%)
30 docs	0.352	0.346 (-1.5%)	0.343 (-2.5%)	0.354 (+0.6%)	0.346 (-1.5%)	0.350 (-0.5%)	0.342 (-2.7%)	0.348 (-1.0%)	0.339 (-3.7%)
100 docs	0.260	0.258 (-0.9%)	0.254 (-2.3%)	0.261 (+0.1%)	0.258 (-0.9%)	0.259 (-0.5%)	0.255 (-2.1%)	0.258 (-1.0%)	0.256 (-1.7%)

thresholds for single-occurrence methods were more aggressive, and produced smaller pruned documents. If the thresholds for single-occurrence methods were increased, the gap between single-occurrence and multiple-occurrences methods might be expected to shrink.

On Testbed I, before content pruning, the CORI and regression-based result-merging algorithms yielded similar results. After content pruning, the regression-based result-merging algorithm was clearly more robust. We had expected the regression-based result-merging algorithm to be more sensitive to pruned documents because the (pruned) documents in the centralized sample database are the training data for learning how to normalize document scores. On Testbed II, in most cases the regression-based algorithm was more effective than the CORI result-merging algorithm. These experimental results are an encouraging sign of stability for the regression-based result-merging algorithm. They also suggest that, for this task, a centralized sample database created from pruned documents is as effective as a database constructed from complete documents.

6.4 Measuring the Effect on Storage Costs

In our experiments, three types of data were affected by pruning: i) the resource selection index, ii) the centralized sample database, and iii) the centralized sample index. Table 6.3 compares the average sizes of these data resources before and after pruning on Testbed I. The percentage figures indicate the reduction in disk space relative to the (unpruned) baseline. Table 6.4 compares the space required to store the sampled documents from the "very long documents" databases in Testbed II (*fr88_merge, patent_merge*) before and after pruning.

These results demonstrate that the pruning of long documents enables dramatic reductions in the storage requirements of three different types of data used in distributed information retrieval.

7. CONCLUSIONS

This paper investigates the effects of pruning documents obtained by query-based sampling on the performance of a distributed IR system. Pruned documents were used to build resource descriptions for resource selection, and a centralized sample database for result merging. Our results demonstrate that pruned documents support resource descriptions, database ranking, and ad-hoc document retrieval accuracies comparable to the accuracies obtained using unpruned documents. However, the amount of storage space was reduced 54-93%, which can be considerable savings when documents are very long.

Our results show that multiple-occurrences methods are slightly better than single-occurrence methods. Single-occurrence methods are sufficient for database ranking, which ignores within-

Table 6.3 The sizes of data resources created from pruned and unpruned documents in Testbed I.

	Full-content	Multiple-occurrences	Single-occurrence
Threshold	N/A	400	100
Resource selection index size	24 MB	11 MB (54.17%)	9 MB (62.50%)
Centralized sample DB size	258 MB	46 MB (82.17%)	19 MB (92.64%)
Centralized sample index size	110 MB	43 MB (60.91%)	25 MB (77.27%)

Table 6.4 The space taken by documents from fr88_merge and patent_merge in the centralized sample database before and after content pruning in Testbed II.

	Full-content	Multiple-occurrences	Single-occurrence
Threshold	N/A	1,600	400
Centralized sample DB size	51 MB	7 MB (86.27%)	2 MB (96.08%)

document term frequency. Regression-based result-merging includes a document retrieval step that is more effective using the within-document term frequency that multiple-occurrences methods provide. However, single-occurrence methods only require about half the space of multiple-occurrences methods, and are surprisingly effective even for the result-merging task. Location-based pruning methods worked well in our tests, even though they ignore frequency, which was a surprise. In general, LUHNM, LUHNS, FIRSTM and RANDM pruning give good and stable performance.

These results provide additional support for the new regression-based result-merging algorithm [9]. This algorithm provided higher baseline results than the CORI result-merging algorithm, and more consistent results across the range of pruning methods.

Future research topics include investigating the effects of pruning in environments with multiple search engine types and different resource selection algorithms, characterizing the bias caused by pruning, and pruning tailored for specific distributed IR tasks.

8. ACKNOWLEDGMENTS

We thank Luo Si and Paul Ogilvie for their help with the research reported here. This material is based on work supported by NSF grant IIS-0096139. Any opinions, findings, conclusions or recommendations expressed in this material are the authors', and do not necessarily reflect those of the sponsor.

9. REFERENCES

[1] J. Allan, J. Callan, M. Sanderson, J. Xu and S. Wegmann. INQUERY and TREC-7. In *Proc. of the Seventh Text Retrieval Conference (TREC-7)*. 1999.

[2] J. Callan, Z. Lu and W. B. Croft. Searching distributed collections with inference networks. In *Proc. of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1995.

[3] J. Callan. Distributed information retrieval. W. B. Croft, editor, *Advances in information retrieval,* chapter 5, pages 127-150. Kluwer Academic Publishers, 2000.

[4] J. Callan and M. Connell. Query-based sampling of text databases. *Transactions on Information Systems*, 19(2), pages 97-130. 2001.

[5] D. Carmel, D. Cohen, R. Fagin, E. Farchi, M. Herscovici, Y. S. Maarek and A. Soffer. Static index pruning for information retrieval systems. In *Proc. of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2001.

[6] N. Craswell, D. Hawking and P. Thistlewaite. Merging results from isolated search engines. In *Proc. of the Tenth Australasian Database Conference*. 1999.

[7] D. Harman, editor. *Proc. of the Third Text Retrieval Conference (TREC-3)*. 1995.

[8] A. Lam-Adesina and G. Jones. Applying summarization techniques for term selection in relevance feedback. In *Proc. of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2001.

[9] S. Luo and J. Callan. Using sampled data and regression to merge search engine results. In *Proc. of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2002.

[10] P. Ogilvie and J. Callan. The Effectiveness of query expansion for distributed information retrieval. In *Proc. of the Tenth International Conference on Information Knowledge Management* (CIKM 2001). 2001.

[11] T. Sakai and K. Sparck-Jones. Generic summaries for indexing in information retrieval. In *Proc. of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2001.

[12] B. Yuwono and D. L. Lee. Server ranking for distributed text retrieval systems on the Internet. In *Proc. of the Fifth International Conference on Database Systems for Advanced Applications (DASFAA)*, pages 41-49. 1997.