Within-Document Term-Based Index Pruning with Statistical Hypothesis Testing

Sree Lekha Thota and Ben Carterette

Department of Computer and Information Sciences
University of Delaware, Newark, DE, USA
lekhat@gmail.com, carteret@cis.udel.edu

Abstract. Document-centric static index pruning methods provide smaller indexes and faster query times by dropping some within-document term information from inverted lists. We present a method of pruning inverted lists derived from the formulation of unigram language models for retrieval. Our method is based on the statistical significance of term frequency ratios: using the two-sample two-proportion (2P2N) test, we statistically compare the frequency of occurrence of a word within a given document to the frequency of its occurrence in the collection to decide whether to prune it. Experimental results show that this technique can be used to significantly decrease the size of the index and querying speed with less compromise to retrieval effectiveness than similar heuristic methods. Furthermore, we give a formal statistical justification for such methods.

1 Introduction

Index pruning is a family of methods for deciding whether to store certain information about term occurrences in documents. It is useful for decreasing index size and increasing query speed, assuming the information lost does not substantially affect retrieval results. Dynamic pruning methods are commonly used to make decisions about cache storage, while static pruning reduces disk storage needs. Static pruning can be either *term-centric*, in which term information is dropped from inverted lists independently of the documents they occur in, or *document-centric*, in which term information is dropped from within documents.

Regardless of the type of pruning, decisions about what to prune are usually made in an ad hoc manner using heuristics. This work presents a method for document-centric pruning derived from the widely-used unigram language modeling approach to retrieval. Like other approaches, we attempt to remove only the terms that are not informative for computing the language model score of a document. Our decisions are based on formal statistical methods that operate under the same modeling assumptions as language modeling: that documents have been sampled term-by-term from some underlying population. Treating the frequency of a term occurrence within a document as an estimate of the proportion of the underlying space that term represents, we can test whether that proportion is equivalent to the proportion in a general background model. If it

is, the term presumably contains no information about the document's relevance to queries including that term.

Specifically, we use the two-sample two-proportion (2P2N) test for statistical significance to determine whether the term frequency within a document is different from its frequency in the background. If we cannot detect significance, we prune the term—we assume it is not informative. The advantage of using statistical significance is not only that it follows from the same modeling assumptions as language models, but also that its errors can be anticipated and controlled in a statistical sense. Thus we hypothesize that we can substantially reduce index size while maintaining greater retrieval effectiveness than heuristic approaches.

2 Previous Work

When reducing index size, we can distinguish between *lossless* techniques and *lossy* techniques. Many lossless methods have been proposed [18, 1, 2, 20]; these are generally compression algorithms that reduce the space required to store an inverted list. These methods are highly effective and are now used in almost all retrieval systems.

Index pruning is a lossy technique: information about the terms and documents is not stored at all. While this can have a large positive effect on space, it can also negatively affect retrieval effectiveness. Thus they should be applied judiciously. Dynamic index pruning techniques are applied during the query time in order to reduce computational cost of query processing. Moffat and Zobel [14] proposed an evaluation technique that uses early recognition of which documents are likely to be highly ranked to reduce costs without degradation in the retrieval effectiveness. Tsegay et al. [19] investigate caching only the pieces of the inverted list that are actually used to answer the query during dynamic pruning. These techniques reduce memory usage, but not disk usage.

Carmel et al. [8] introduced the concept of static index pruning technique to the information retrieval systems. They present a term-centric approach in which for each term in the index only the top k postings are retained. The main idea behind this method is to use the search engine's ranking in order to evaluate the importance of each inverted list and determine which entries can be removed from the index. Each term in the index is submitted as a query to the search engine and from the resulting document set for pruning. The term is removed from the document D if it is not present in the top k portion of the ranked result set from the search engine.

Büttcher and Clarke [5, 6] presented a document centric approach: the decision about whether the term's posting should be present or not depends on its rank by a score computed within a document rather than the posting's rank within its term posting list. For each document D in the corpus, only the postings for the top k terms in the document are kept in the index.The terms are ranked based on their contribution to the document's Kullback-Leibler divergence from the rest of the collection.

In 2005, de Moura et al. [13] proposed a locality based static pruning method which is a variation of Carmel's method that aims at predicting what set of terms may occur together in queries and using this information to preserve common documents in the inverted lists of these term. A boosting technique for Carmel's static index pruning has been proposed by Blanco and Barreiro [4] in which they use the probabilistic-based scoring function (BM25) instead of the tf-idf method and address some features like updating of the document lengths and the average document length in the pruned inverted file which are not considered in the original model. More recently, Nguyen [15] presented a posting based approach which is a generalization of both document-centric and term-centric approaches.

3 Pruning Using the Two-Sample Two-Proportion Test

As described in Section 1, our pruning method is derived from the unigram language model. We start by showing the derivation, then describing the statistical method we will use. We then refine the method to account for possible errors.

3.1 Language Modeling

The basic idea for our method is derived from the query-likelihood retrieval model. Language modeling [21, 12] is one of the most effective and widely-used retrieval models. In the unigram language model, documents are modeled as term-by-term samples from some underlying population. They are ranked by the probability of sampling the query Q from the multinomial "bag of words" representing a document D, i.e. by the value of $P(Q|D)$. This is estimated as:

$$P(Q|D) = \prod_{q_i \in Q} P(q_i|D) \simeq \sum_{q_i \in Q} \log\left(P(q_i|D)\right)$$

where

$$P(q_i|D) = \frac{tf_{q_i,D}}{|D|}$$

and $tf_{q_i,D}$ is the number of times term q_i occurs in the document D, and $|D|$ is the total number of terms in the document. Since this probability could be zero if just one query term fails to occur in the document, the model is smoothed with a background model based on the full collection of documents [21], which is also modeled as a sample from an underlying space:

$$P(q_i|D) = \lambda \frac{tf_{q_i,D}}{|D|} + (1-\lambda)\frac{ctf_{q_i}}{|C|}$$

where λ is a smoothing parameter, ctf_{q_i} is the total number of occurrences of q_i in the entire collection C, and $|C|$ is the total number of terms in the collection. We are agnostic about the modeling assumptions that lead to a particular choice

of form or value of λ; the Dirichlet prior is a common approach that has been shown to work well in practice [21].

From the above equation, we can see that when the ratio of document term count to document length is exactly equal to the ratio of collection term frequency to the total collection term count, the two λ-scaled ratios cancel out and the score of the document depends only on the collection term frequency:

$$\frac{tf_{q_i,D}}{|D|} = \frac{ctf_{q_i}}{|C|} \Rightarrow P(q_i|D) = \frac{ctf_{q_i}}{|C|}$$

Since the document's final score does not depend on the frequency of such a term, that information can be pruned with no penalty to retrieval effectiveness.

In general, we cannot expect that the two ratios computed from data will be exactly equal, even if they actually are equivalent in the underlying term populations from which D and C have been sampled. The nature of sampling means that the ratios are only estimates of the "true" underlying values, and may be higher or lower randomly but within well-defined ranges. Thus we need a way to test whether the two ratios are equivalent in the *statistical* sense of falling within a given confidence interval.

3.2 The Two-Sample Two-Proportion Test

The two-sample two-proportion (2N2P) test is a statistical procedure for testing the hypothesis that two proportions are equal given two estimates of those proportions calculated from two different samples [11, chapter 6]. We start by computing the difference between two proportions. Because those proportions are based on samples, they have some variance. When their difference is not exactly zero, the variance may still be high enough that we can consider them effectively equivalent. Dividing the difference between the two proportions by a standard error produces a normally-distributed test statistic Z that we can use to make a decision about whether to consider the two proportions different.

The value of the Z statistic is calculated using the formula

$$Z = \frac{\frac{x_1}{n_1} - \frac{x_2}{n_2}}{E}$$

where n_1, n_2 are the sample sizes, x_1, x_2 are the number of observed occurrences, and E is the standard error of the difference in proportions. The standard error is calculated as:

$$E = \sqrt{P\left(1 - P\right)\left(\frac{1}{n_1} + \frac{1}{n_2}\right)}$$

where

$$P = \frac{x_1 + x_2}{n_1 + n_2}$$

Z has an approximately standard normal distribution, and thus to determine whether the difference in proportions is significant we check the probability of observing a value of Z and higher (and/or $-Z$ and lower, depending on the type of test) in a normal distribution with mean 0 and standard deviation 1. If that probability is less than some pre-selected value α, we reject the hypothesis that the proportions are the same. α is frequently chosen to be 0.05, which corresponds to $|Z| \approx 2$ in a two-sided test or $|Z| \approx 1.65$ in a one-sided test. In Figure 1, for instance, we would reject the hypothesis that the two proportions are equal (or that x_2/n_2 is greater) if we calculate $Z > 1.65$.

3.3 Static Index Pruning Using the 2N2P Test

We will use the above described statistical method to make pruning decisions. In our method, we calculate the value of the Z statistic of each term in a document. This value is calculated by using the document length and the collection length as the sample sizes and the ratios of frequency of the word in the document to the document length and the frequency of the word in the entire collection to the collection length as the proportions. Based on the value of the term's Z statistic, we decide whether to keep the word in the index or to drop it. The value of the Z statistic gives us the significance of the term to the document.

$$Z = \frac{\frac{tf_{q_i,D}}{|D|} - \frac{ctf_{q_i}}{|C|}}{E}$$

where $tf_{q_i,D}$ is the frequency of the term in the document, $|D|$ is the length of the document, ctf_{q_i} is the frequency of the term in the entire collection, $|C|$ is the total number of terms in the entire collection and E is the standard error. The standard error is calculated using the following formula,

$$E = \sqrt{P\left(1-P\right)\left(\frac{1}{|D|} + \frac{1}{|C|}\right)}$$

where

$$P = \frac{tf_{q_i,D} + ctf_{q_i,D}}{|D| + |C|}$$

Note that we are using the same assumptions as the unigram language model: that document and collection are sampled term-by-term from an underlying space, and the term proportions are thus estimates of their true occurrence.

We next choose a threshold value of Z to denote the significance level needed to keep information about a term in the index, i.e. we choose a value for Z *a priori* and store only those terms whose calculated value is greater than this value. Note that choosing different thresholds is equivalent to choosing different significance levels α; in Figure 1 we have chosen a threshold of 1.65, corresponding to $\alpha = 0.05$ in a one-sided test. As the threshold increases (significance level decreases), the size of the pruned index decreases.

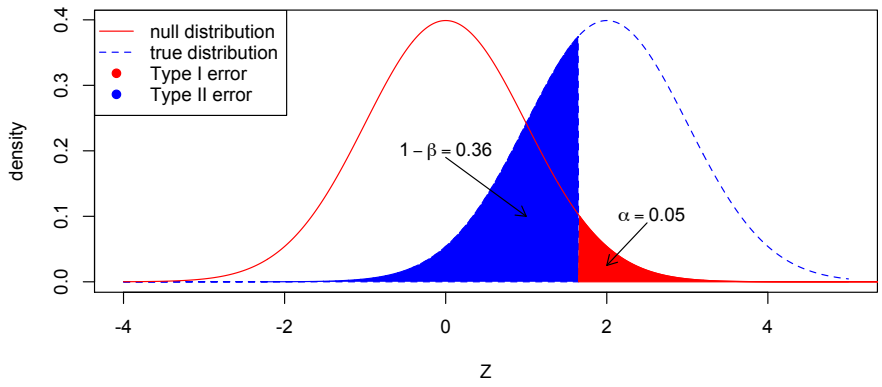

Fig. 1. Illustration of statistical power. If the null hypothesis is true (red normal density curve), there is a 5% probability of a Type I error of not pruning a non-informative term (red shaded region for a one-sided test). If the null hypothesis is *not* true and the true value of the Z-statistic is 2 (blue normal density curve), there is a 36% probability of a Type II error of pruning an informative term (blue shaded region) and consequently only 64% probability of correctly keeping that term.

Therefore, the value of Z for a term gives us the level of importance of the term to the meaning of the document. Only the terms that are meaningful to the document are added to the index; the remaining terms are discarded. Note also that the number of terms pruned from each document is different and depends on their informative content rather than the length of the document. The resulting size of the index depends on the number of postings that are significant enough, based on the Z value we specify, to prune from each document.

Note that the stored values of the document lengths and collection statistics must not be modified for our method to work. If the test tells us to prune a term from a document, only its document-level *tf* value is pruned from the index. All other information about the document and collection remains unchanged, including document length $|D|$, collection frequency *ctf*, and collection length $|C|$. Even the fact that a pruned term appears in a document must still be stored (to handle the difference between a term with $tf = 0$ and one with $tf/|D| = ctf/|C|$), though this can be done with a minimum of overhead. If any of these values changed, the derivation in Section 3.1 above would no longer work.

3.4 Statistical Power of the 2N2P Test

Using the results of a statistical hypothesis test to make a decision always has some chance of resulting in an incorrect action. In our case, we may incorrectly decide to keep a term that is not meaningful to the document (a Type I error of finding a significant difference when one does not exist), or we may incorrectly

decide to prune a term that is meaningful to the document (a Type II error of failing to find a significant difference when one exists). Using different thresholds for Z controls the Type I error: the lower Z is, the more likely we are to prune terms, and therefore Type I errors become more likely.

Our method is meant to determine when term counts do not need to be stored to maintain retrieval effectiveness, as we showed in Section 3.1. We can continue to store them if we are willing to accept the cost of the disk space and query processing time. This means that Type I errors are relatively cheap. Type II errors are substantially more expensive: once we've decided to prune a term, we cannot use any information about it in calculating document scores. If we were wrong to prune it, it may significantly and negatively impact retrieval performance. Therefore we would like to be able to control the probability of Type II errors as well as Type I errors when pruning terms from documents.

Type II error rates are inversely related to *statistical power*. Power is usually denoted β, and the expected Type II error rate is $1-\beta$. Power analysis [11] allows us to use known quantities such as document length and collection size along with a desired Type I error rate and effect size (described below) to determine when it is best to prune a term.

Figure 1 illustrates Type I and Type II errors. If the null hypothesis is true, the Z-statistic will be drawn from the normal density function centered at zero (colored red). If the threshold for rejection is $\alpha = 0.05$ ($Z \approx 1.65$), then there is a 5% chance of a Type I error. But if the null hypothesis is *not* true, the Z-statistic will be drawn from some other distribution. In this example, we suppose that the "true" value is 2, and the observed value will be sampled from a variance-1 normal distribution centered at 2 (colored blue). The probability of a Type II error, then, is the probability that the observed value is *less than* the threshold. If it is, we would fail to reject the null hypothesis, even though it is not true.

To implement power analysis, we first define an estimated *effect size* that we consider large enough to be meaningful. Effect size, denoted h, is a dimensionless quantity computed from the proportions and P:

$$h = \frac{\frac{tf_{q_i,D}}{|D|} - \frac{ctf_{q_i}}{|C|}}{\sqrt{P(1-P)}}$$

A loose guide to interpretation of effect size is that an effect size between 0 and 0.2 is considered "small", between 0.2 and 0.4 is "moderate" and greater than 0.4 is "strong". We could choose to keep terms only if the effect size is strong, i.e. only if the estimated ratios are substantially different. Or we could choose to keep terms with small effect sizes on the assumption that Type I errors are "cheap" and it takes a lot of evidence for us to decide to prune a term.

Once we have chosen an effect size, we calculate the value of α (equivalently, the Z statistic threshold) that would result in finding a significant difference with probability β. This is done by solving the following equation for α_D.

$$\Phi\left(\Phi^{-1}(\alpha_D) - h\sqrt{\frac{1}{|D|} + \frac{1}{|C|}}\right) - \beta = 0$$

To understand this, consider each component in turn: $\Phi(Z)$ is the standard normal cumulative density function, i.e. the area under the standard normal density curve from Z to ∞. It always has a value between 0 and 1. Its inverse $\Phi^{-1}(\alpha_D)$ is therefore the threshold for Z that would result in a Type I error rate of α_D. We shift that value by effect size h scaled by a function of the total evidence we have (measured by $|D|$ and $|C|$), then calculate the probability of observing a Z of that value or greater in a standard normal distribution. In Figure 1, $\alpha_D = 0.05$, $\Phi^{-1}(\alpha_D) \approx 1.65$, $h\sqrt{\frac{1}{|D|} + \frac{1}{|C|}} \approx 2$, and $\Phi(1.64 - 2) \approx 0.64$. This is the power achieved when $\alpha_D = 0.05$.

There is no closed-form solution for α_D, so we solve it with linear search. Once we have the value of α_D, the corresponding Z_D can be found using normal distribution tables or by another application of the quantile function. We then apply pruning exactly as in Section 3.3: when the Z_D statistic is greater than that computed by power analysis, the term is kept; otherwise it is pruned.

The practical effect of this is essentially that each document has its own threshold for pruning, and that threshold is based on two parameters: desired effect size h and desired power β to detect that effect size. So we trade one parameter (a global Z threshold) for two that give us a local threshold for each document Z_D. Furthermore, since effect size and Type II error rate are monotonically related, we can effectively reduce the parameter space to a single parameter—desired power β. Increasing the power parameter results in lower local Z_D thresholds, which in turn results in fewer terms being pruned.

4 Experimental Results

4.1 Data

For empirical analysis, we used the GOV2 collection of 25,183,256 documents as our corpus to index. GOV2 has been used to experiment on efficiency as part of the TREC Terabyte track [9, 10, 7]. The queries we used to evaluate effectiveness are title queries from topic numbers 701 through 750 developed for the 2006 TREC Terabyte track.

4.2 Building the Index

We used the Indri retrieval engine [17] for indexing and query processing. We implemented all pruning methods in Indri. We used the Krovetz stemmer and a stopword list of 420 words that is included in the Lemur source distribution.

For calculating the value of Z at index time, we refer to a complete unpruned index of the dataset in order to obtain the term frequency in the document, the term frequency in the entire collection, document lengths, and the collection size. The Indri code is modified such that before each term is added to the index, this calculated value of Z is compared to the desired value (submitted as a parameter at runtime) and is added to the index only if it is higher compared to the desired value. We did not alter stored document lengths, collection lengths, and collection term frequencies.

Index size	k	MAP	Prec@10
100%	-	0.2642	0.5106
40.85%	100	0.1437	0.4601
58.54%	200	0.1675	0.4786
65.84%	250	0.1817	0.4800
76.89%	350	0.2130	0.4917
90.39%	1000	0.2519	0.5107

Table 1. Pruning using KL-Divergence.

Index size	Z	MAP	Prec@10
100%	0	0.2642	0.5106
42.61%	50	0.1531	0.4578
56.61%	30	0.1662	0.4745
68.98%	10	0.1978	0.4900
75.32%	5	0.2136	0.4978
92.1%	1.69	0.2527	0.5106

Table 2. Pruning using the 2N2P test.

4.3 Baseline

We compare to Büttcher and Clarke's document-centric KL-divergence method [6] described in Section 2. We calculate the KL-divergence score of each of the terms in the document, and the top k terms are retained in the document while the others are pruned. The following formula is used to calculate the KL-divergence scores of the terms in the document:

$$Score_{DCP}(t_i) = P(t_i|D) \log \left(\frac{P(t_i|D)}{P(t_i|C)} \right)$$

where $P(t_i|D)$ and $P(t_i|C)$ are calculated as in Section 3.1 above. Again, Indri is modified such that only the top k terms in each document are stored in the index and the rest are pruned. For different values of k, different index sizes are obtained.

4.4 Evaluation

We calculate the size of a pruned index as a percentage of the complete unpruned index. Our goal is to test whether retrieval speed and effectiveness are substantially affected by pruning using the 2N2P tests, and to compare those tests to the baseline. We evaluate effectiveness by mean average precision (MAP) and mean precision at rank 10 (prec@10) over the 50 Terabyte queries. We evaluate retrieval speed by the total time it takes to process those queries.

4.5 Results

Table 1 shows the results of the KL-Divergence method. The various index sizes are obtained by repeating the experiments with increasing values of k, which is the number of terms stored from each document. The MAPs obtained at different index sizes are shown. Table 2 shows the results of varying a global Z-statistic for the 2N2P test to produce different index sizes. Table 3 shows the results using 2N2P power analysis with desired effect size $h = 0.2$ and varying power β. Note that index sizes are not identical across the tables because there is no way to guarantee that each method will result in the same number of pruned postings. We have chosen parameter values that produce roughly equal index sizes.

Index size	MAP	Prec@10
100%	0.2642	0.5106
43.23%	0.1799	0.4345
58.21%	0.1831	0.4837
68.24%	0.2021	0.4900
77.54%	0.2329	0.4946
92.16%	0.2600	0.5070

Table 3. Pruning using power analysis.

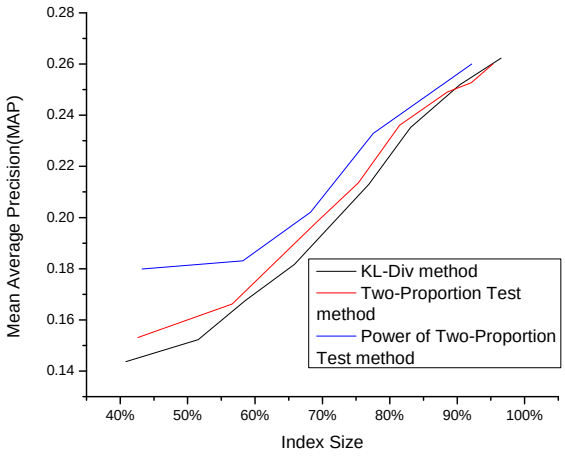

Fig. 2. Index size vs. MAP for the three pruning methods.

In all cases MAP and prec@10 decrease with index size, but it is clear from the results that, given an index size, the statistical hypothesis testing method presented in this paper provides a small increase in effectiveness. Furthermore, MAP scores obtained using power analysis show substantial improvement over both methods.

Since we cannot guarantee that the methods will produce the same-size index, effectiveness results are summarized graphically in Figure 2. Here we can see that the 2N2P methods produce nearly uniformly better results across index sizes, and the gain from using power analysis is strong..

Query times are illustrated in Figure 3. The two methods based on 2N2P are both slightly faster than the KL-Divergence method, though they are not substantially different from each other: with only 50 queries, the variance is high enough that these results are not significant. We did not have any specific hypothesis about query times; we present these results out of interest.

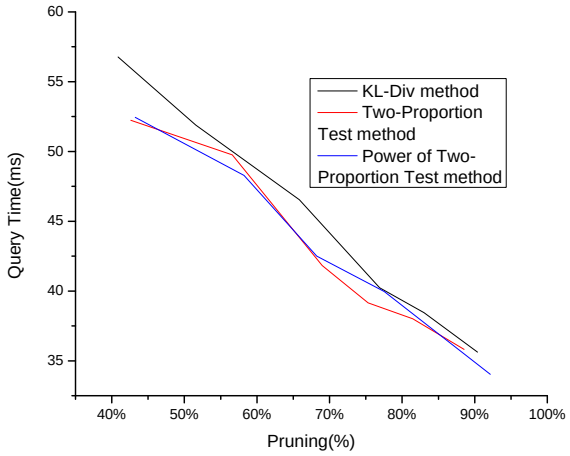

Fig. 3. Index size vs. query processing time for the three pruning methods.

5 Conclusions and Future Work

We have presented a within-document term based index pruning method that uses formal statistical hypothesis testing. In this method, the terms in the document which have the least effect on the score of the document are pruned from the index, thus reducing its size with little compromise in theory on the effectiveness of the retrieval. The significance of the terms is calculated by using the Z statistic value from a two-sample two-proportion test that document term frequency is equivalent to collection term frequency.

We implemented two different approaches of this technique, one of which uses a constant threshold of Z irrespective of the document length, the other calculating a threshold of Z for each document based on its length using power analysis. From our experimental results, these methods not only decreased the index size but also were relatively successful in maintaining the performance of the system compared to the KL-Divergence method.

Our results are based on formal statistical analysis rather than heuristics, and derived from the same assumptions as the query-likelihood language model. Thus they suggest why static pruning methods work: they use evidence about documents and collections to eliminate information from the index that is irrelevant for scoring the document against queries. We believe similar statistical approaches could be used to prune indexes optimally for other retrieval methods, including BM25; an interesting future direction may be statistical pruning of more specific information such as term positions for use with more complex models such as Indri's inference network model.

References

1. Vo Ngoc Anh and Alistair Moffat. Inverted index compressed using word-aligned binary codes. *Inf. Retr.*, 8(1):151–166, January 2005.
2. Vo Ngoc Anh and Alistair Moffat. Pruned query evaluation using precomputed impacts. In *Proceedings of SIGIR*, 2006.
3. Leif Azzopardi and D. E. Losada. An efficient computation of the multiple-bernoulli language model. In *Proceedings of ECIR*, pages 480–483, 2006.
4. Roi Blanco and Alvaro Barreiro. Boosting static pruning of inverted files. In *Proceedings of SIGIR*, 2007.
5. Stefan Büttcher and Charles L. A. Clarke. Efficiency vs. effectiveness in terabyte-scale information retrieval. In *Proceedings of TREC*, 2005.
6. Stefan Büttcher and Charles L. A. Clarke. A document-centric approach to static index pruning in text retrieval systems. In *Proceedings of CIKM*, 2006.
7. Stefan Büttcher, Charles L. A. Clarke, and Ian Soboroff. The TREC 2006 Terabyte track. In *Proceedings of TREC*, 2006.
8. D. Carmel, D. Cohen, R. Fagin, E. Farchi, M. Hersovici, Y. Maarek, and A. Soer. Static index pruning for information retrieval systems. In *Proceedings of SIGIR*, pages 43–50, 2001.
9. Charles L. A. Clarke, Nick Craswell, and Ian Soboroff. Overview of the TREC 2004 Terabyte track. In *Proceedings of TREC*, 2004.
10. Charles L. A. Clarke, Falk Scholer, and Ian Soboroff. The TREC 2005 Terabyte track. In *Proceedings of TREC*, 2005.
11. Jacob Cohen. *Statistical Power Analysis for the Behavioral Sciences*. Routledge Academic, 2nd edition, 1988.
12. W. Bruce Croft and John Lafferty, editors. *Language Modeling for Information Retrieval*. Springer, 2003.
13. Edleno S. de Moura, Celia F. dos Santos, Daniel R. Fernandes, Altigran S. Silva, Pavel Calado, and Mario A. Nascimento. Improving web search efficiency via a locality based static pruning method. In *Proceedings of WWW*, 2005.
14. Alistair Moffat and Justin Zobel. Self-indexing inverted files for fast text retrieval. *ACM TOIS*, 14(4):349–379, October 1996.
15. L. T. Nguyen. Static index pruning for information retrieval systems: A posting-based approach. In *Proceedings of LSDS-IR, CEUR Workshop*, pages 25–32, 2009.
16. Michael Persin, Justin Zobel, and Ron Sacks-Davis. Filtered document retrieval with frequency-sorted indexes. *JASIS*, 47(10):749–764, October 1996.
17. Trevor Strohman, Donald Metzler, Howard Turtle, and W. Bruce Croft. Indri: A language-model based search engine for complex queries. In *Proceedings of the International Conference on Intelligent Analysis*, 2005.
18. Andrew Trotman. Compressing inverted files. *Inf. Retr.*, 6:5–19, January 2003.
19. Yohannes Tsegay, Andrew Turpin, and Justin Zobel. Dynamic index pruning for effective caching. In *Proceedings of CIKM*, 2007.
20. Ian H. Witten, Alistair Moffat, and Timothy C. Bell. *Managing Gigabytes*. Morgan Kaufman, 1999.
21. ChengXiang Zhai and John Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM TOIS*, 22(2):179–214, April 2004.