WITHIN-DOCUMENT TERM-BASED INDEX PRUNING TECHNIQUES WITH STATISTICAL HYPOTHESIS TESTING

by

Sree Lekha Thota

A thesis submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Master of Science in Computer Science

Fall 2010

WITHIN-DOCUMENT TERM-BASED INDEX PRUNING

TECHNIQUES WITH STATISTICAL HYPOTHESIS TESTING

by

Sree Lekha Thota

Approved: _____
Ben Carterette, Ph.D.
Professor in charge of thesis on behalf of the Advisory Committee

Approved: _____
Errol Lloyd, Ph.D.
Chair of the Department of Computer and Information Sciences

Approved: _____
Michael Chajes, Ph.D.
Dean of the College of Engineering

Approved: _____
Charles G. Riordan, Ph.D.
Vice Provost for Graduate and Professional Education

ACKNOWLEDGMENTS

The satisfaction that accompanies the successful completion of any task would be incomplete without introducing the people who made it possible and whose constant guidance and encouragement crowns all the efforts with success.

It gives me immense pleasure to express my gratitude and indebtedness to my advisor Professor Ben Carterette, for his invaluable and inspiring guidance throughout the progress of the thesis work. I would like to thank all my professors at University of Delaware who helped me broaden my knowledge.

I would also like to thank my family especially my sister Rekha and Brother-in-law Adithya for providing me with unconditional support throughout my masters. I would like to mention my friend Spandana for being so helpful and supportive.

I take this opportunity to express my deepest gratitude to all those who helped me directly or indirectly at various stages of my thesis work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Static index pruning methods have been proposed to reduce the index size of information retrieval systems while retaining the effectiveness of the search. 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 for pruning inverted lists derived from the formulation of unigram language models for retrieval. This method is based on the statistical significance of term frequency ratios. Using the two-sample two-proportion (2P2N) test, the frequency of occurrence of a word within a given document is statistically compared 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 time with less compromise to retrieval effectiveness than similar heuristic methods. We also implemented static index pruning algorithm that uses the retrievability of the documents decide whether to remove or keep them in the index, along with the statistical hypothesis testing method. The retrievability is calculated using the document entropy which is in turn calculated using the entropies of each of the terms in the document. It is observed from the experimental results that the performance of the retrieval system is improved by this hybrid algorithm. Furthermore, a formal statistical justification for such methods is also given.

Chapter 1

INTRODUCTION

Search engines make use of inverted indexes for obtaining an efficient query evaluation. Inverted indexes contain inverted files [14] which give an association between terms and documents in the collection. For a large text corpus, the inverted file is too large to fit into memory of the search engine and is also difficult to maintain. Therefore, search engines typically use different compression techniques in order to compress these index files. The primary goal is to increase the efficiency in terms of query response time while sustaining the effectiveness in terms of the ranking quality.

Large amount of work has been done in the field of index compression and many techniques have been introduced in the past. These techniques can be mainly categorized into two complementary approaches which are *lossless compression* and *lossy compression*. The lossless compression [15, 16, 17, 20] uses data compression techniques to compress index data and thus reduce the disk space used for storing the index files without losing information. On the contrary, lossy compression techniques achieve very high compression ratios by reducing the size of the index by discarding the presumably less informative content and keeping only the important information. The usage of a lossless compressed index reduces the size of the index but does not have the same effect on the time taken to process the inverted lists during the query processing time. Therefore, the efficiency of the system cannot be increased by using lossless compression alone.

Index pruning is a general 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 methods are used to make decisions about disk storage. 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. In this work we present a method for document-centric pruning derived from the widely-used unigram language modeling approach to retrieval. Like other approaches, we attempted to remove only the terms that are not informative for computing the language model score of a document. The decisions are based on formal statistical methods that operate under the same modeling assumptions as language modelling: 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 the term represents, it can be tested if 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 significance is not detected, the term is pruned assuming that it is not informative. The advantage of using statistical significance is that it not only follows from the same modelling assumptions as language models, but also that the errors it results in can be anticipated and controlled on a statistical sense. Thus, it can be hypothesized that

index size and querying time can be substantially reduced while maintaining greater retrieval effectiveness than heuristic approaches.

Through this method, the size of the index can be pruned to up to 70% of the original index for the entire collection while maintaining the important information from the documents, thus giving high-quality results for most search queries. Also, the amount of disk space used to store the index is also considerably reduced which results in improved query processing performance at the cost of a minor decrease in retrieval effectiveness.

In addition to this method, we also implement a static index pruning method based on the retrievability of the documents. All the documents which are less retrievable by the IR system are removed from the index, thus reducing the index size with least effect on the efficiency. In this method the retrievability score for each document in the collection is calculated based on the entropy of each term in the document. According to the desired pruning level, a threshold for the retrievability score of the document is set and all the documents with score below the threshold are removed from the index. We use a hybrid of this retrievability based method to remove the documents from the collection and then remove the terms which do not contain information about the document's relevance from the other documents, thus reducing the size of the index significantly while maintaining the retrieval effectiveness.

1.1 Thesis Organization

The report is organized as follows. Chapter 2 gives a background of the related work in the area of index pruning. The various dynamic and static index pruning techniques proposed so far are discussed in this section.

Chapter 3 gives an overview of the baseline retrieval system that is used for the experiments and also gives a detailed description of the different new

pruning algorithms. This includes description of the language modelling, the two sample two proportionality test and an iterative approach.

Chapter 4 describes a method which is a combination of the new algorithm and uses the document's retrievability. This chapter comprises of the derivation of the formula for retrievability score for the documents followed by a description of the hybrid method implemented.

Chapter 5 gives a description of the datasets used, the experimental setup and a brief explanation of the baseline method and the evaluation method adopted. The experimental results obtained are also presented in this chapter with graphs comparing the different methods.

Finally, the report is concluded in Chapter 6 with some more ideas for future work.

Chapter 2

PREVIOUS WORK

Many lossless index compression techniques were proposed in the past. These techniques mainly depend on the fact that the inverted lists are essentially lists of numbers and without compression, each number takes up the same amount of space. Also these term frequencies in the inverted index follow a skewed distribution. Therefore, various data compression methods like Delta encoding, etc., can be used for encoding the index information, thus reducing the size of the index by about one-tenth. But the index size is still too big and these compression techniques do not reduce the time required to process the inverted lists. Therefore, using lossless index compression techniques alone will not suffice.

2.1 Dynamic Index Pruning

Dynamic index pruning techniques have been introduced which are applied during the query time in order to reduce the computational cost for the query processing. These methods are particularly of importance when large, static, collections are being distributed on relatively slow read-only media. A. Moffat and J. Zobel [7], 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. In this method, the query response time for conjunctive Boolean queries and for informal ranked queries is reduced, at little cost in terms of storage, by the inclusion of an internal index in each inverted list. The methods described in this paper show that both memory space and processing time can be simultaneously reduced.

The information retrieval systems store the entire inverted list in the cache while only a small portion of the lists are processed. Yohannes Tsegay et al. [10] investigated caching only the pieces of the inverted lists that are actually used to answer the query during dynamic pruning. The LRU cache model is examined in this paper, in addition to the use of cost-aware caching eviction strategies that use attributes besides the removing of a cache element to determine what item is to be removed. The authors also put forward a new dynamic pruning scheme for impact ordered inverted lists in which postings are stored in sorted blocks of decreasing term-document impact score.

2.2 Static Index Pruning

The concept of static index pruning technique to information retrieval systems was introduced be Carmel et. Al [2]. The method presented in the paper is a term-centric approach in which for each term in the index only the top *k* postings are retained in the index. 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. Ideally, this technique will preserve the terms set large enough to give a good approximation of the ranking scores for the top results for each of the queries submitted to the system. The paper looks both into uniform and term-based methods. In uniform pruning there is a fixed threshold and all index entries whose contribution to relevance scores is below the threshold are removed from the index whereas in the term-based pruning the cut-off threshold is determined for each term and is therefore different from term to term.

Buttcher and Clarke [4] use an approach similar to the method described by Carmel et. al, but in contrast to their approach, it creates a pruned index much smaller than the original index. In this method, instead of one index, two separate indices were created. One of them, a pruned index, holds postings for the most frequent terms in the collection while the other index has the postings for the entire collection. The pruned index only has, for every term, only the *k* documents in which the term's impact on the BM25 score of that document was the highest. For each term *T* among the *n* most frequent terms in the collection and each document, the documents score for the term *T* as the query term is computed and only the *k* documents with highest score are kept in the index. The values of the *n* and *k* are chosen such that the pruned index is small enough to keep in the memory while the bigger index is stored on disk. The two indices are used in parallel during query processing.

In contrast to the term-centric approaches above, Buttcher and Clarke also presented a document centric approach. In this approach the decision whether the term's posting should be present or not depends on its rank within the document it refers to instead of the posting's rank within its term posting list. For each document *D* in the corpus, only the postings for the top k_D terms in the document are kept in the index and the others are dropped. The terms are ranked based on its contribution to the documents Kullback-Leibler divergence from the rest of the collection. A pseudo-relevance feedback step is performed for every document based on the KL divergence scores at the indexing time and only postings for the top k_D feedback terms extracted from the document index are retained in the index. The pseudo-relevance feedback predicts the set of query terms for which the document *D* can be in the top documents. Therefore, only those terms appearing in the queries are stored in the index.

Moura et al. [6] 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 uses this information to preserve common documents in the inverted lists of these term, that is, besides top entries of each postings for each term, it also preserves entries that are in the top list of other related terms. Some types of queries such as conjunctive queries and phrases require that the pruning method preserves index entries for documents that occur in the inverted lists of different terms, that is, documents where two query terms occur together. Carmel's method accounts only for the individual terms but not the phrases and therefore important information may be pruned away from the index. This paper proposes a method to determine significant sentences and retaining them in the index.

A boosting technique for the Carmel's Static index pruning has been proposed by Blanco and Barreiro [9] in which they use the probabilistic-based scoring function (BM25) instead of the *tf-idf* method and addresses 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. This method improves the performance with respect to preserving the document ranking and was successful in increasing the precision values at some pruning level and under certain conditions.

More recently, Nguyen [5] presented a posting based approach which is a generalization of both document-centric and term-centric approaches. This approach ranks all the postings and keeps only a subset of top ranked ones. The rank of the posting depends on various factors such as its rank in its inverted list, its rank in its document, its weighing score, the term weight and the document weight. This method evaluates the postings and assigns a usefulness value to each of them. The pruned index is built based on these values. For a given desired level

of pruning, posting entries are selected based on their usefulness values and added into the pruned index until the pruned index size reaches its limit.

Blanco and Barreiro [22] presented a pruning technique which is based on a probabilistic model of information retrieval. The Probabilistic Ranking Principle is used as a decision criterion over which posting list entries are to be pruned. This method is not aimed at maintaining the top results returned by the IR system but it defines a pruning criterion that relies on the goodness of the estimation of three probabilities. This approach follows a term by term mechanism.

Anh and Moffat [11] presented a non-static pruning technique in which only the index is pruned based in the impact-order of the terms. This is a dynamic query pruning technique which eliminates redundant part of the exhaustive evaluation of the ranked queries. This method makes use of the impact-sorted indexes. From the experimental results it is observed that compared to exhaustive evaluation, this method reduced the amount of computation performed, reduced the amount of memory required for accumulators, reduced the amount of data transferred from the disk and at the same time allow performance guarantees in terms of precision and mean average precision.

All the methods described so far use heuristics in order to decide which postings are to be dropped from the index. In this work, we present a document-centric method similar to the method presented by Buttcher and Clarke [4] but instead of depending on ad hoc heuristics, we use formal statistical methods for deciding which terms are to be pruned from the index. As mentioned earlier, the advantage of using statistical significance is that, errors it results in can be anticipated and controlled on a statistical sense. In addition to this, we also try to improve the compression ratios by removing some documents from the index which are not retrievable.

Chapter 3

PRUNING USING TWO-SAMPLE TWO-PROPORTION TEST

As described in Chapter 1, this pruning method is derived from the unigram language model. This chapter starts with the derivation and then explains the statistical method used. Then the method is refined to better account for possible errors.

3.1 Language Modeling

The basic idea for this method is derived from the query-likelihood retrieval model. Language modeling [15, 16] 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 of terms. 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 can be estimated as:

$$P(Q|D) = \prod_{q_i \in Q} P(q_i|D)$$

where

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

and $tf_{qi,D}$ 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 [12], which is also modelled as a sample from an underlying space:

$$P(Q|D) \propto \sum_{q_i \in Q} log\left(\lambda \frac{tf_{q_i,D}}{|D|} + (1 - \lambda)\frac{ctf_{q_i}}{|C|}\right)$$

where λ is a smoothing parameter, ctf_{qi} 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. The log-probability is used because it does not change the relative ranking of documents by this score, and it is easier to compute sums of logs of small probabilities than their products. We are agnostic about the modelling 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 [12].

From the above equation, we can see that when the ratio of document term count to document length and the ratio of term count in the entire collection to the total collection term count are equal, the two ratios cancel out and the score of the document only depends on the collection term frequency. Therefore, removing the term from the document has no effect on its score. Information about it can be pruned from the index with no penalty to retrieval effectiveness. However, we cannot expect that those two ratios will be exactly equal even when 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 *statistically* equivalent.

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 an approximately 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,

$$Z = \frac{\left({}^{x_1}/_{n_1} - {}^{x_2}/_{n_2}\right)}{E}$$

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:

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

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 is checked. 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 3.1, 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.

$$Z = \frac{\left({tf_{q_i,D}}/{|D|} - {ctf_{q_i}}/{|C|} \right)}{E}$$

where $tf_{qi,D}$ is the frequency of the term in the document, $|D|$ is the length of the document, ctf_{qi} 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,

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

where

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

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 3.1, a threshold of 1.65 is chosen, corresponding to $\alpha = 0.05$ in a one-sided test. As the threshold value increases (significance level decreases), the size of the pruned index decreases.

Therefore, the value of Z for a term gives us the level of importance of the term in the document and only the terms which represent the document are added to the index and the remaining terms are discarded. Also, the number of terms pruned in index in each document is different and depends on the informative content in the document 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 specified, in every document.

Note that the stored values of the document lengths and collection statistics must not be modified for this 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 $|C|$. If any of these values are 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, it may be incorrectly decided 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 it may be incorrectly decided 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 it is 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 shown 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, it is desired 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 by 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 3.1 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 case, 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, it would fail to reject the null hypothesis, even though it is not true.

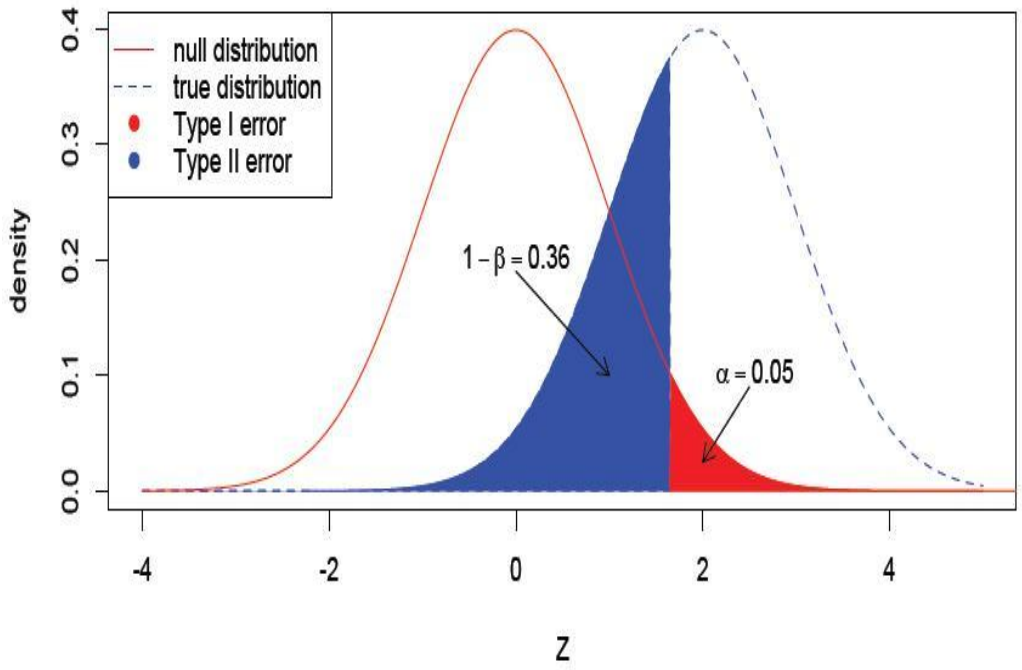

Figure 3.1: **Illustration of statistical power**. If the null hypothesis is true (red normal density curve), there is a 5% probability of a Type I error of not pruning a non informative term (red shaded region for a one-sided test). If the null hypothesis is not true and the true value of the *Z*-statistic 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.

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

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

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.

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

Φ is the normal cumulative density function, so $\Phi^{-1}(\alpha_D)$ is 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 Figure 3.1, α_D = 0.05, $\Phi^{-1}(\alpha_D) \approx 1.65$, $h\sqrt{\frac{1}{|D|} + \frac{1}{|C|}} \approx 2$, and Φ (1.64 - 2) ≈ 0.64. This is the power achieved when α_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 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 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 directly related, we can effectively reduce the parameter space to a single parameter - desired effect size. Increase in the effect size parameter results in lower local Z_D thresholds, which in turn results in fewer terms being pruned.

3.5 Iterative 2N2P Test

In the statistical methods used in this work, a significance score is calculated for each term in the document and the decision to prune it or keep it in the index is made based on this value. In certain cases, terms which contain information about the document might be incorrectly decided to prune away. It is important to retain these terms in the index in order to maintain the search quality. We used an iterative method in order to add some of these terms back into the index.

In this method, the statistical 2N2P test method is used same as described in the previous section, that is, depending on the desired pruning level, we assume a threshold for Z and all the terms with Z values more than this threshold are pruned away from the index. But in some cases, some of the informative terms can have a higher value of Z and these might also be pruned away in the process. In

this method, after pruning the value of *Z* with a certain threshold, some of the words with *Z* value higher than the threshold are added back to the index. This value of *Z* can be selected as a range depending on which words we would like to add back to index. This range is some range above the threshold which we used for pruning. Generally, the range starting from the threshold to the higher value is taken depending on the desired level of pruning but we can also take a different range of Z and add back the terms with these values back into the index. This can also be continued over many iterations, that is, using a threshold value of *Z* and a range of values to add words back to the index thus leaving the index with terms having a wide range of *Z* values instead of just the ones below the threshold.

This method is implemented as follows. In the first iteration, we have the complete unpruned-index of the dataset and the index built using the statistical pruning techniques discussed previously by using a value of *Z* as the threshold. In the second iteration, we make use of these two indexes, we prune the pruned index further using a smaller value of *Z* as the threshold while we add the words back from the unpruned-index with a different range of *Z* value. For the next iteration, we can use the index obtained from the previous iteration and the pruned index from the first iteration as the two indexes and repeat the same procedure as in second iteration. If this is continued, as the number of iterations progresses, we obtain an index which has terms with a variety of *Z* values. These terms represent the terms which are informative having variant *Z* values due to some reason and mislabelled to be non-informative terms by the basic 2N2P test. Iterative methods help us add these terms back into the index.

Chapter 4

STATIC INDEX PRUNING USING THE RETRIEVABILITY MEASURE

The information retrieval system provides a means to access the information from the enormous number of documents available. In a way, the retrieval system has an effect on the documents that we can access from the collection. Depending on the behavior of the retrieval system, some documents are more retrievable than the others. In this chapter, we present a way to recognize the documents which are not retrievable or less retrievable and remove them from the collection in order to reduce the index size.

4.1 Retrievability

Retrievability of a document for a given information retrieval system is defined as the ease with which the document can be retrieved from the underlying collection. Due to various reasons, some documents in the corpus are never retrieved or shown in the ranked list for any query by the IR system. Since these documents are never, or very rarely, retrieved by the search engine, they are never shown in the ranking list of any query and therefore removing these documents from the index has very little effect on the performance of the search engine.

We tried a few basic methods for finding such documents which are not retrievable. The idea was to submit all possible combinations of terms as queries to the IR system and remove the documents which are not present in the top *k* ranked lists of any query. Since it is impractical to obtain all the queries possible, we experimented with the subset of million queries. After getting the top *k* ranked lists for all the queries, we combine the lists and drop all the documents from the index which are not present in this list.

An improvement over this basic method that we implemented was, to calculate and assign a retrievability score for each of the documents. This score is calculated by submitting all possible queries to the system and adding the number of query ranked lists that the document was shown in by the IR system. All the documents with retrievability score less than a predefined threshold were pruned from the index.

4.2 Estimating Retrievability using Information Entropy

4.2.1 Information Entropy. Information entropy is a measure of the uncertainty associated with a random variable. It measures the homogeneity of the examples. The entropy of a discrete random variable X which is divided into n sets $\{x_1, x_2, \ldots, x_n\}$ is defined as

$$H(X) = -\sum_{i=1}^{n} P(x_i) \log P(x_i)$$

where $P(x_i)$ is the probability distribution function of the random variable.

Consider for example, the tossing of a coin with known but not necessarily fair probabilities of coming up head or tails. If the coin is fair, the entropy of the unknown result of the next toss is maximized. This is the situation of maximum uncertainty as it is most difficult to predict the outcome of the next toss. However, there is less uncertainty if the coin is not fair and the probabilities of heads and tails are p and q. The reduced uncertainty is given by lower entropy. The extreme case is when we have a double-headed coin in which case tails never come up and there is no uncertainty. The entropy is zero and we do not get any information from each coin toss. This is shown in the graph below.

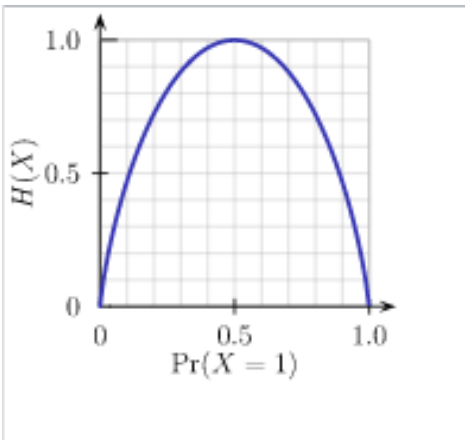

Figure 4.1: Entropy *H(X)* (i.e. the expected surprisal) of a coin flip, measured in bits; graphed versus the fairness of the coin *Pr(X=1)*.

4.2.2 Document entropy. Each term in a corpus can also be considered as a random variable which is divided into different sets of documents with different values. Therefore from the definition of entropy described above, the entropy of a term in a corpus gives a measure of the homogeneity of the term in the corpus. Therefore, a term with high entropy value is supposed to be spread across evenly in the entire corpus and no useful information can be obtained from such terms. A document which has a high proportion of such terms can also be considered less informative and hence less retrievable. These documents can be safely removed from the corpus decreasing the index size while maintaining the search efficiency. These documents can be identified by calculating the entropies of all the terms in the corpus and using these values to calculate retrievability score for the document itself. All the documents whose retrievability score is not in the desired range can be pruned away.

The entropy values for each of the terms in the collection can be calculated by modifying the equation for entropy for the random variable. The

probability that a term q_i occurs in a document D is given by the following formula,

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

where $tf_{qi,D}$ is the number of times the term q_i occurs in the document D and ctf_{qi} is the number of times the term occurs in the whole collection.

The entropy of a term q_i is given by the equation

$$H(q_i) = -\sum_{D \in C} P(D|q_i) log P(D|q_i)$$

Substituting the value of probability in the above equation we get the entropy of the term to be

$$H(q_i) = -\sum_{D \in C} \left(\frac{tf_{q_i,D}}{ctf_{q_i}} log \frac{tf_{q_i,D}}{ctf_{q_i}} \right)$$

After the entropies of all the terms are calculated, the importance score of each of the documents is calculated by summing all the entropy values of each of the terms in the document. Also different documents have different lengths which might affect the score of the document. This can be avoided by normalizing the value by dividing the summation with the document length.

$$Score_D = -\frac{1}{|D|} \sum_{q_i \in D} \left(tf_{q_i,D} \sum_{D \in C} \left(\frac{tf_{q_i,D}}{ctf_{q_i}} log \frac{tf_{q_i,D}}{ctf_{q_i}} \right) \right)$$

From this equation, the documents which have a more terms with high entropy value have a high score compared to the documents containing terms with low entropy. The documents with high score shows a high language entropy which

suggests that the document contains information about a lot of different topics and therefore is unlikely to be useful to any topics. Therefore, the documents entropy can be considered as a good parameter for measuring the retrievability of the documents.

4.3 Index Pruning using Retrievability

As discussed above, the document entropy values can be used as a measure of their retrievability. The documents with high score are considered less retrievable whereas the documents with a low score are considered more retrievable. We implemented a method for index pruning which uses the retrievability scores obtained from the document entropies in conjunction with the statistical hypothesis testing methods described in the previous chapter.

In this method, we start by constructing an unpruned-index of the dataset. From this index, the entropy values of all terms in the collection are calculated. These values are in turn used to calculate the retrievability scores of the documents by using the formulas derived in the previous section. At this point a threshold value for the importance scores is decided based on the desired pruning level. Now, a second indexing run is performed in which all the documents which have the importance score higher than the threshold are removed from the index and all the other documents are retained and are further processed using the statistical 2N2P test method. As the decision of whether to keep or drop the documents is made during the indexing time, all the information regarding the document and the collection is preserved.

Chapter 5

EXPERIMENTAL RESULTS

5.1 Data

For the empirical analysis, we used the WT2G and GOV2 text collections as datasets. WT2G is a part of webtrack of TREC '99 and GOV2 is a part of the TREC Terabyte collection. We took the queries used for evaluating the system from topics 401 to 450 in the ad-hoc task of TREC 8 and topics 701 to 750 from the TREC terabyte track. We used the titles of these topics as the queries and the Indri platform to index the collection and rank the queries. The following table gives a brief description of the datasets and the queries used.

Table 5.1: Datasets and Queries used

Dataset	No. of Documents	Queries Used
WT2G	247491	TREC ad-hoc task 401-450
GOV2	25183256	TREC Terabyte track 701-750

5.2 Building the Index

We used the Indri retrieval engine [19] for indexing and query processing. All index pruning methods in Indri are implemented. The Krovetz stemmer and the stopword list of 420 words that is included in the Lemur distribution is used for indexing.

For calculating the value of Z, we used the complete un-pruned index of the dataset in order to obtain the statistics such as the term frequency in the

document, the term frequency in the entire collection, documents 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 and is added to the index only if it is higher than the desired value. The stored document lengths, collection lengths, and collection term frequencies are not altered.

5.3 Baseline

We compare the results to those of Buttcher and Clarke's document-centric KL-divergence method [3] described in Chapter 2. The KL-divergence score of each of the terms in the document is calculated and the top *k* terms are retained in the document and 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 Chapter 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.

5.4 Evaluation

We calculate the size of a pruned index as a percentage of the complete un-pruned index. The 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 the effectiveness by mean average precision (MAP) calculated over 50 queries and precision@10 and the retrieval speed by the total time it takes to process 50 queries.

Precision is defined as the fraction of the documents that are relevant to the user's information need. This is given by dividing the number of relevant

document among the retrieved document by the number of retrieved documents. Precision takes all retrieved documents into account and can also be evaluated at a given cut-off rank considering only the topmost results returned by the system. This measure is called precision at n. The Mean Average Precision (MAP) for a set of queries is defined as the mean of the average precision scores for each query. For our evaluation, we submit a set of 50 queries to the retrieval system and calculate the MAP and Precision@10 for the results obtained from the results. These values are compared to the MAP and Precision@10 values obtained with the KL-Divergence method with index of the same size.

The average time taken for query processing is also considered for the evaluation of the pruning method. The query processing time is calculated as the average over the time taken for the system to process 40,000 queries. This can also be done by submitting each of the queries separately to the search engine, every time a query is submitted, the search engine has to open the index, retrieve the results and close it again. The opening and closing of the index takes up considerable amount of time and therefore the average query processing time is higher when the queries are submitted individually to the system.

5.5 Results

The following sections give the results using the different index pruning methods implemented. In all cases MAP decreases with index size, but it is observed 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 score obtained using power analysis show substantial improvement over both KL-Divergence method and the 2N2P test method. It is also observed from graphs that the iterative and the hybrid methods give a slight improvement in the results.

It is observed that the basic retrievability methods discussed in section 4.1, performed well when tried individually but when combined with the statistical hypothesis test methods, they did not give expected results as the collection statistics are changed when we drop the documents from the collection and it is important to maintain the collection information for the statistical hypothesis test method to be effective.

Following are the results using the different pruning techniques presented in this report.

5.5.1 KL-Divergence Method. Table 5.2 shows the results of the KL-Divergence method using the WT2G dataset and Table 5.3 shows the results using the GOV2 dataset. The various index sizes are obtained by repeating the experiments with increasing values of *k*, which is the number of terms sorted from each document. The MAPs obtained at different index sizes are shown.

Table 5.2: Results with Pruning using KL-Div method on WT2G Dataset

Index Size(% of Complete Index)	k	MAP	Prec@10
100%	-	0.2948	0.4220
95.87%	1500	0.2924	0.422
92.40%	1000	0.2876	0.426
85.32%	600	0.2716	0.426
82.35%	500	0.2671	0.424
80.48%	450	0.2637	0.424
75.72%	350	0.2507	0.418
68.56%	250	0.2313	0.404
63.65%	200	0.2238	0.404
57.45%	150	0.2077	0.402

Table 5.3: Results with Pruning using KL-Div method on GOV2 Dataset

Index Size(% of Complete Index)	k	MAP	Prec@10
100%	-	0.2642	0.5106
96.50%	1500	0.2623	0.5107
90.39%	1000	0.2519	0.5107
83.03%	500	0.2352	0.4974
76.89%	350	0.213	0.4917
65.84%	250	0.1817	0.48
58.54%	200	0.1675	0.4786
51.57%	150	0.1523	0.47
40.85%	100	0.1437	0.4601

5.5.2 The Two-Sample Two-Proportion Test. Tables 5.4 and 5.5 show the results using the global *Z*-statistic method for different index sizes using the WT2G and GOV2 datasets respectively.

Table 5.4: Results with Pruning using Two-Proportion Test on WT2G Dataset

Index Size(% of Complete Index)	Z value	MAP	Prec@10
100%	0	0.2948	0.4220
98.09%	0.5	0.2945	0.422
96.15%	1	0.2943	0.424
93.24%	1.69	0.2879	0.42
91.99%	2	0.2888	0.424
81.61%	5	0.2735	0.422
70.38%	10	0.2566	0.414
63.75%	15	0.2408	0.41
58.73%	20	0.2228	0.39
51.94%	30	0.1953	0.376
43.84%	50	0.1673	0.412

Table 5.5: Results with Pruning using Two-Proportion Test on GOV2 Dataset

Index Size(% of Complete Index)	Z value	MAP	Prec@10
100%	0	0.2642	0.5106
95.30%	1	0.26	0.51
92.10%	1.69	0.2527	0.5106
88.53%	2	0.249	0.5001
81.45%	3	0.2361	0.4978
75.32%	5	0.2136	0.4978
68.98%	10	0.1978	0.49
56.61%	30	0.1662	0.4745
42.61%	50	0.1531	0.4578

5.5.3 Statistical Power of the 2N2P Test. Tables 5.6 and 5.7 show the results with the WT2G and GOV2 datasets using the 2N2P power analysis with desired effect size $h = 0.2$.

Table 5.6: Results with Pruning using Power of Two-Proportion Test on WT2G Dataset

Index Size(% of Complete Index)	MAP	Prec@10
100%	0.2948	0.4220
83.06%	0.2887	0.4156
79%	0.2845	0.41
63.89%	0.2562	0.4145
57.30%	0.2389	0.4026
44.60%	0.1907	0.4056

Table 5.7: Results with Pruning using Power of Two-Proportion Test on GOV2 Dataset

Index Size(% of Complete Index)	MAP	Prec@10
100%	0.2642	0.5106
92.16%	0.26	0.507
77.54%	0.2329	0.4946
68.24%	0.2021	0.49
58.21%	0.1831	0.4837
43.23%	0.1799	0.4345

5.5.4 Iterative 2N2P Test Method. Tables 5.8 and 5.9 show the results with the WT2G and GOV2 datasets using the iterative 2N2P test method.

Table 5.8: Results with Pruning using Iterative 2N2P test on WT2G Dataset

Index Size(% of Complete Index)	MAP
100%	0.2948
95.67%	0.2946
90.61%	0.2882
80.11%	0.2729
72.02%	0.2601
64%	0.2469
51.88%	0.2106
29.48%	0.1114
23.44%	0.0798

Table 5.9: Results with Pruning using Iterative 2N2P test on GOV2 Dataset

Index Size(% of Complete Index)	MAP
100%	0.2642
96.1 %	0.2630
91.2%	0.2525
83.5%	0.2463
74.92%	0.2142
66.49%	0.1971
55.3%	0.1659

5.5.5 Hybrid Method. Tables 5.10 and 5.11 show the results obtained by using the 2N2P and Entropy methods together.

Table 5.10: Results with Pruning using Hybrid method on WT2G Dataset

Index Size(% of Complete Index)	MAP
100%	0.2948
95.04%	0.2929
91.47%	0.2913
84.97%	0.2832
80.14%	0.2772
78.66%	0.2764
65.23%	0.2590
47.9%	0.2119

Table 5.11: Results with Pruning using Hybrid method on GOV2 Dataset

Index Size(% of Complete Index)	MAP
100%	0.2642
95.39%	0.2628
90.12%	0.2543
85.72%	0.2500
77.6%	0.2289
65.38%	0.2011
56.14%	0.1757

5.5.6 Comparison of the Pruning Methods. We assess the efficiency of different pruning methods by plotting the Index size vs. MAP graph for each method. Figure 5.1 shows the graph for the WT2G dataset and figure 5.2 shows the graph using the GOV2 dataset. It can be observed from both the graphs that all the new pruning techniques presented in this report give an improvement over the KL-Divergence method used as the baseline for comparison. Also the Power of 2N2P test method has the better performance as compared to all the other methods. The combination entropy and 2N2P method gives an improvement in the results over the basic 2N2P and the Iterative 2N2P methods and also performs better than the Power of 2N2P test method in certain cases.

Figure 5.3 shows the index size to average query processing time relation using 50queries and Figure 5.4 shows the index size to query processing time using 40,000queries. It was observed that the performance of statistical methods used in this work is faster than that using the complete index and comparable to the baseline method in terms of the time taken per query. It can be seen that with the decrease in index size, there is a decrease in the query processing time.

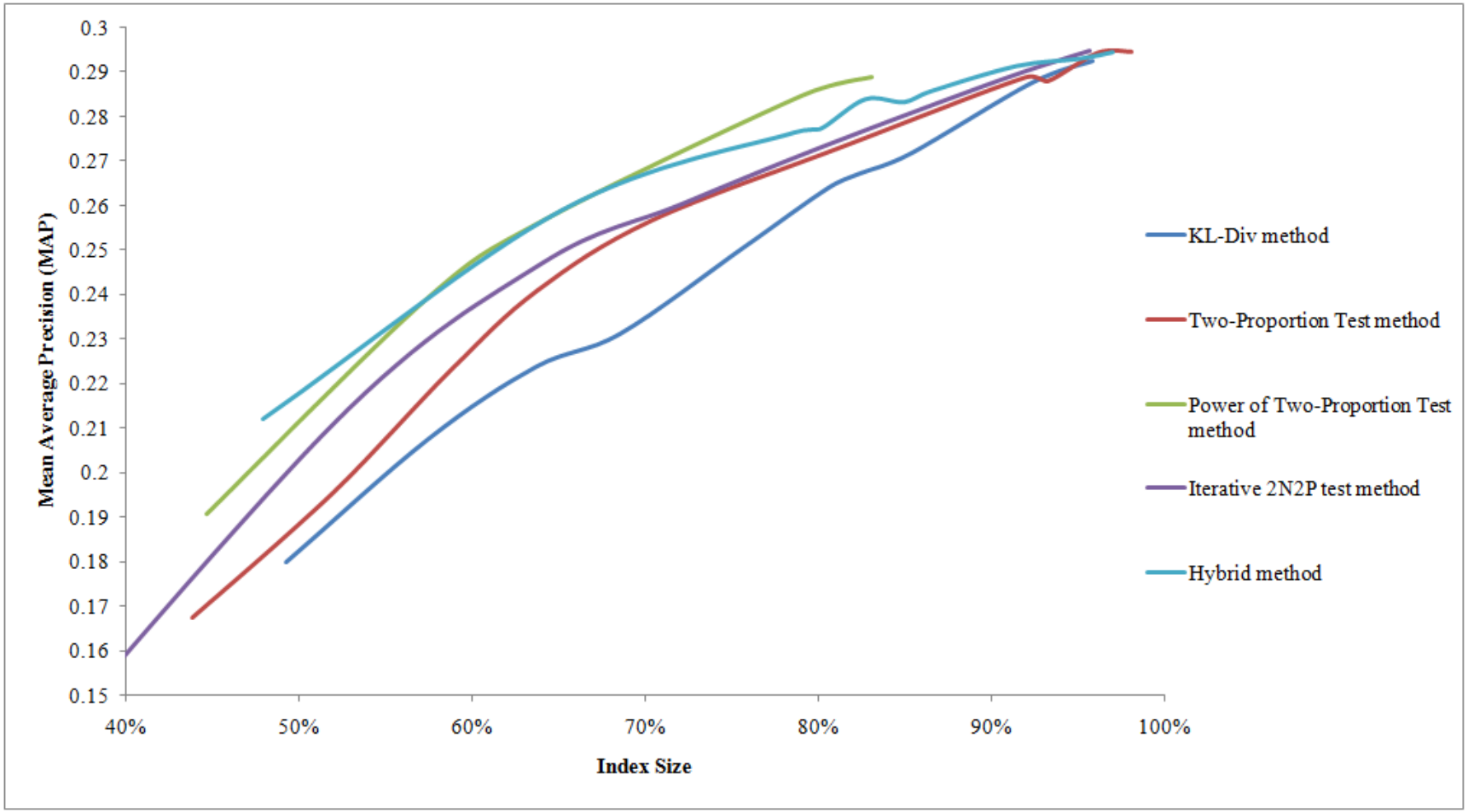

Figure 5.1: Index Size vs. MAP for KL-Div, Two-Proportion Test, Power of Two-Proportion Test, Iterative 2N2P test and Hybrid methods using WT2G results

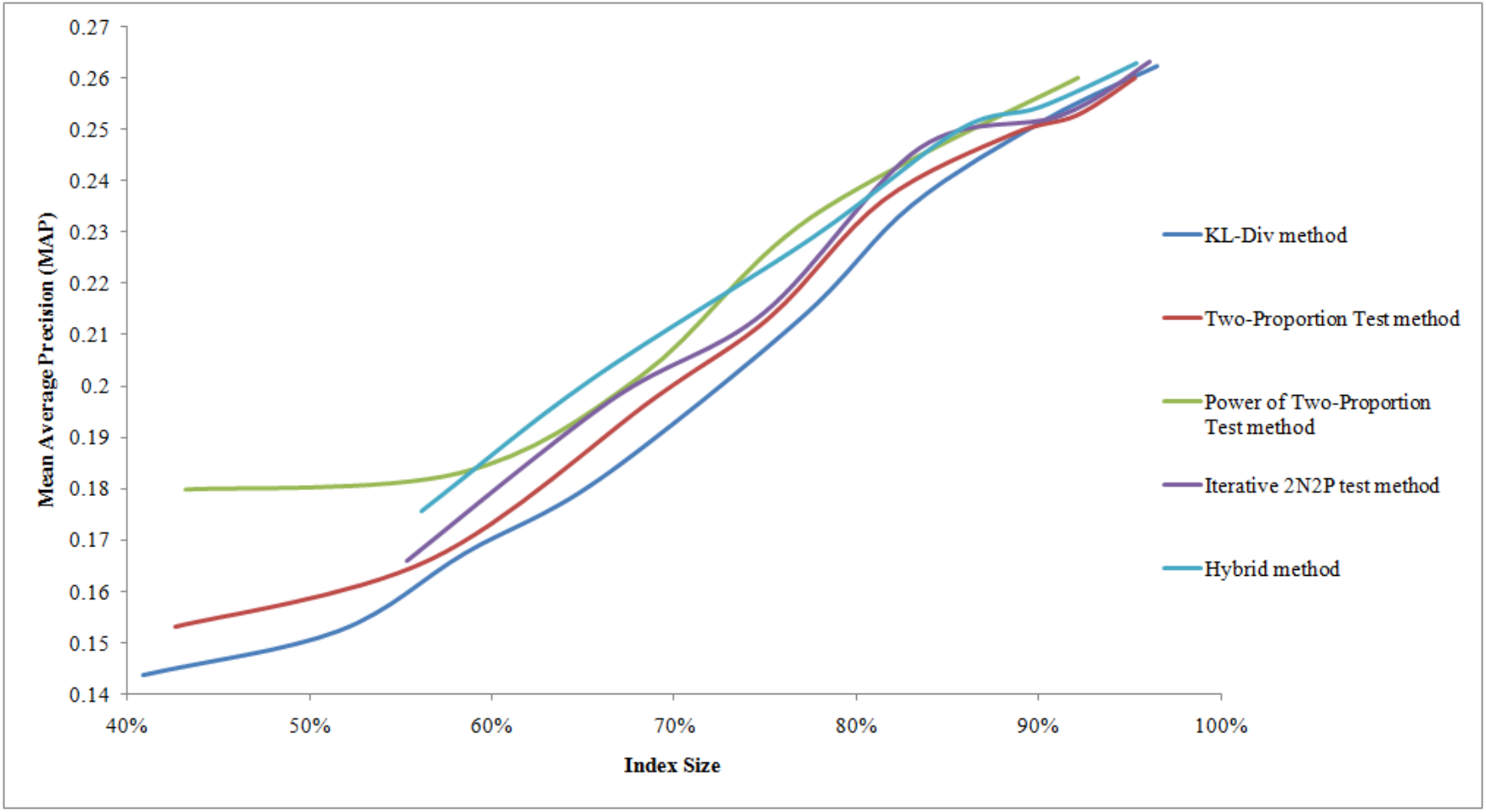

Figure 5.2: Index Size vs. MAP for KL-Div, Two-Proportion Test, Power of Two-Proportion Test, Iterative 2N2P test and Hybrid methods using GOV2 results

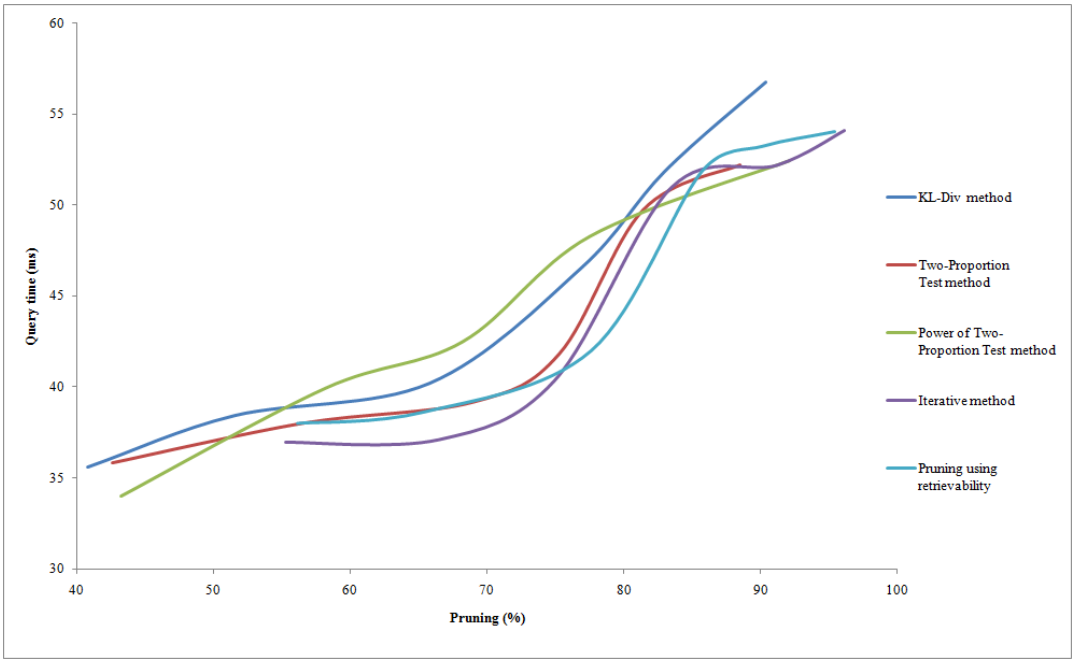

Figure 5.3: Index size vs. average time per query using 50 queries for the KL-Div, Two-Proportion Test, Power of Two-Proportion Test, Iterative and Hybrid methods

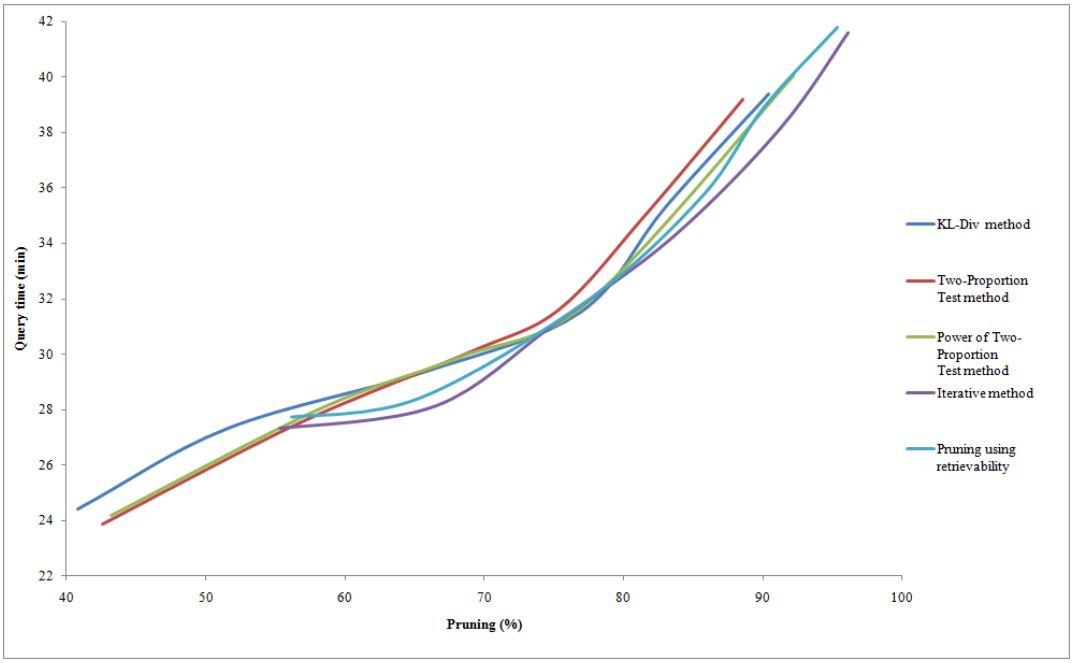

Figure 5.4: Index size vs. Query processing time using 40,000 queries for the KL-Div, Two-Proportion Test, Power of Two-Proportion Test, Iterative and Hybrid methods

Chapter 6

CONCLUSIONS AND FUTURE WORK

We have presented a within-document term based index pruning method which uses statistical hypothesis testing. In this method, the terms in the document which have the least or no effect on the score of the document are pruned from the document and thus reducing the size of the index without compromising on the effectiveness of the retrieval. The significance of the terms is calculated by using the *Z* statistic value of the term in the document with respect to the collection.

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. We also presented an iterative approach in which some of the pruned words are added back to the index.

We also combined our new algorithm with a static index pruning algorithm using the retrievability measure which removes the documents which are less retrievable based on its importance score.

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 non-informative terms.

The corpus may contain some spam documents and these documents can be removed from the index. Many spam detection techniques such as trust ranking can be used to determine if the page is spam or not. Our future work involves exploring the methods to remove the spam documents from the corpus and thus from the index.

Also, we are trying to find some more ways in which we can add some of the terms which are pruned by our method but are actually important to the document, back to the index in order to improve the efficiency of retrieval.

REFERENCES

[1] W. Bruce Croft, Donald Metzler, Trevor Strohman. Search Engines: Information Retrieval in Practice. *Addison Wesley*.

[2] D. Carmel, D. Cohen, R. Fagin, E. Farchi, M. Herscovici, Y. Maarek, and A. Soer. Static Index Pruning for Information Retrieval Systems. *In Proceedings of the 24th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 43-50, 2001.

[3] S. Buttcher and C. L. A. Clarke. Efficiency vs. Effectiveness in Terabyte-Scale Information Retrieval. *In Proceedings of the 14th Text Retrieval Conference*, Gaithersburg, USA, November 2005.

[4] S. Büttcher and C. L. A. Clarke. A Document-Centric Approach to Static Index Pruning in Text Retrieval Systems. *In CIKM*, 2006.

[5] L.T. Nguyen. "Static Index Pruning for Information Retrieval Systems: A Posting-Based Approach." *In Proc. LSDS-IR, 2009*, pp. 25--32. *CEUR Workshop Proceedings*, Vol. 80, ISSN 1613-0073.

[6] Edleno S. de Moura, Celia F. dos Santos, Daniel R. Fernandes, Altigran S. Silva, Pavel Calado, Mario A. Nascimento. Improving Web Search Efficiency via a Locality Based Static Pruning Method. *In proceedings of WWW 2005*, May 10-14, 2005.

[7] A. Moffat and J. Zobel. Self-Indexing Inverted Files for Fast Text Retrieval. *ACM Transactions on Information Systems.* 14(4): 349-379, October 1996.

[8] M. Persin, J. Zobel, and R. Sacks-Davis. Filteres Document Retrieval with Frequency-Sorted Indexes. *Journal of the American Society for Information Science*, 47(10): 749-764, October 1996.

[9] Roi Blanco, Alvaro Barreiro. Boosting Static Pruning of Inverted Files. *In proceedings of SIGIR '07 July 23-27, 2007*, Amsterdam.

[10] Yohannes Tsegay, Andrew Turpin, Justin Zobel. Dynamic Index Pruning for Effective Caching. *In the Proceedings of CIKM'07.*

[11] V. N. Anh and A. Moffat. Pruned Query Evaluation Using Precomputed Impacts. *In Proceedings of the 29th ACM SIGIR Conf. on Research and Development in Information Retrieval*, Seattle, USA, 2006.

[12] Jacob Cohen, Statistical Power Analysis for the Behavioral Sciences. Lawrence Erlbaum Associates, Inc.

[13] Zhai et al. A Study of Smoothing Methods for Language Models

[14] Justin Zobel , Alistair Moffat, Inverted files for text search engines, ACM Computing Surveys (CSUR), v.38 n.2, p.6-es, 2006

[15] Andrew Trotman. Compressing Inverted Files. ACM Information Retrieval 2003.

[16] Vo Ngoc Anh , Alistair Moffat, Index compression using fixed binary codewords, Proceedings of the fifteenth Australasian database conference, p.61-67, January 01, 2004, Dunedin, New Zealand.

[17] Vo Ngoc Anh , Alistair Moffat, Inverted Index Compression Using Word-Aligned Binary Codes, *Information Retrieval*, v.8 n.1, p.151-166, January 2005

[18] Croft W. Bruce, Lafferty, John. Language Modeling for Information Retrieval. ISBN: 978-1-4020-1216-7. 2003.

[19] L. Azzopardi and D. E. Losada. An Efficient Computation of the Multiple-Bernoulli Language Model. *In Proceedings of the 28th European Conference on Information Retrieval (ECIR 2006), pages 480{483, London, UK, April 2006.*

[20] I. H. Witten, A. Moffat, and T. C. Bell. *Managing Gigabytes*. Morgan Kaufman, San Francisco, 1999.

[21] Trevor Strohman, Donald Metzler, Howard Turtle, W. Bruce Croft. Indri: a language-model based search engine for complex queries (2005). *In Proceedings of the International Conference on Intelligent Analysis.*

[22] Lei Zheng, Ingemar J. Cox. Entropy-Based Static Index Pruning. *In the proceedings of European Conference for Information Retrieval (ECIR 2009).*