# Entropy-Based Static Index Pruning

Lei Zheng and Ingemar J. Cox

University College London
Adastral Park Campus
Suffolk, IP5 3RE, United Kingdom
lei.zheng@ucl.ac.uk, ingemar@ieee.org

**Abstract.** We propose a new entropy-based algorithm for static index pruning. The algorithm computes an importance score for each document in the collection based on the entropy of each term. A threshold is set according to the desired level of pruning and all postings associated with documents that score below this threshold are removed from the index, i.e. documents are removed from the collection. We compare this entropy-based approach with previous work by Carmel *et al.* [1], for both the Financial Times (FT) and Los Angeles Times (LA) collections. Experimental results reveal that the entropy-based approach has superior performance on the FT collection, for both precision at 10 (P@10) and mean average precision (MAP). However, for the LA collection, Carmel's method is generally superior with MAP. The variation in performance across collections suggests that a hybrid algorithm that incorporates elements of both methods might have more stable performance across collections. A simple hybrid method is tested, in which a first 10% pruning is performed using the entropy-based method, and further pruning is performed by Carmel's method. Experimental results show that the hybird algorithm can slightly improve that of Carmel's, but performs significantly worse than the entropy-based method on the FT collection.

## 1 Introduction

An inverted index is a data structure that is commonly used to implement information retrieval (IR) [2]. An inverted index can be thought of as a table in which the rows represent all of the terms (words) presented in the collection, and the entries in a row, known as postings, point to the documents containing the term. It is apparent that for large collections, the inverted index may become extremely large. Thus, a major line of research focuses on reducing the index size, which is referred to as pruning. Index pruning can be either dynamic or static. Dynamic pruning [1] decides during query processing, whether certain terms or document postings are worth adding to the accumulated document scores, and whether the ranking process should continue or stop. By contrast, static pruning [1] removes entries from the index in advance (of any query), therefore reducing the index size. Static and dynamic pruning can complement each other.

Much of the work regarding static index pruning has focused on developing various pruning algorithms for either (i) removing less important terms from the index [3], i.e. eliminating rows from the index table, or (ii) removing less important postings from the index [1,4], i.e. sparsifying the index table. In this paper, we present a new entropy-based approach to prune the inverted index. Our method is to decide whether all the postings for a given document should remain in the index. Removing all the postings pointing to a particular document is equivalent to removing a document from the collection. The decision on which documents to remove is made according to an entropy-based importance score. After removing less important documents in a collection, we could still apply further pruning techniques, such as techniques presented in [1,3,4]. A hybrid approach incorporating Carmel's pruning is also described.

The remainder of the paper is organized as follows: Section 2 revisits Carmel's static pruning technique. Section 3 discusses our entropy-based index pruning. The experiments and results are presented in Section 4. The paper ends with a conclusion and future work section.

## 2 Carmel's Static Index Pruning

Carmel *et al.* [1] introduced the concept of static index pruning and described a pruning algorithm that removes less important postings from an inverted index. The algorithm is called top-$k$ pruning, and involves two parameters, i.e. $k$ and $\epsilon$, to be described shortly. The procedure [1,3] to select which postings to remove from the index is performed on a per-term basis. For each term in the lexicon (vocabulary), the algorithm computes the term's contribution to the documents containing the term using the score function of the retrieval system. After that, the algorithm retrieves the term's $k$th highest posting score $z_t$ and sets a threshold $\tau_t = \epsilon \cdot z_t$, where $\epsilon$ is a parameter that can be used to control the pruning rate. Finally, all the postings with scores lower than $\tau_t$ are deemed unimportant postings, and removed from the term's posting list.

A variant of the algorithm is called $\delta$-top pruning. For each term in the lexicon, $\delta$-top pruning removes the postings for which the scores are lower than $\delta$ times the highest score $z'_t$. We point out that the nature of top-$k$ pruning and $\delta$-top pruning are the same. The only difference is whether the pruning threshold is determined by the $k$th highest score $z_t$ or the highest score $z'_t$.

## 3 Entropy-Based Static Index Pruning

The information entropy of a discrete random variable $X$ with possible values $\{x_1, x_2, \cdots, x_n\}$ is defined as

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

where $P(x_i)$ is the probability distribution function (PDF) of the random variable $X$. The concept of entropy gives some idea of how random the variable $X$ is. Consider an example of a two-side coin. If the probability of the occurrence of either side is 1/2, the entropy achieves a maximum, simply because there is maximum uncertainty (information content) in the outcome of the toss. However, if the probability for one side is 1/4 and 3/4 for the other side, the value of entropy becomes smaller. In such case, the uncertainty decreases and the variable is more predictable.

We consider each term $t_i$ in the lexicon as a random variable. The probability, $P_j(t_i)$, of term $t_i$ occurring in document $d_j$, where $j$ ranges from 1 to $n_d$ ($n_d$ denotes the number of documents in the collection), is given by

$$P_j(t_i) = \frac{tf(d_j)}{tf(c)} \quad (j = 1, 2, \cdots, n_d) \tag{2}$$

where $tf(d_j)$ is $t_i$'s term frequency in document $d_j$, and $tf(c)$ denotes $t_i$'s term frequency in the whole collection $c$. Under such definition, we have the entropy of a term $t_i$ as

$$H(t_i) = -\sum_{j=1}^{n_d} P_j(t_i)\log(P_j(t_i)) = -\sum_{j=1}^{n_d} \frac{tf(d_j)}{tf(c)} \log\left(\frac{tf(d_j)}{tf(c)}\right) \tag{3}$$

After the entropy of each term $t_i$ is computed, our entropy-based score of document importance is defined as

$$S(d) = \frac{1}{l_d}\sum_{t_i\in d} tf(t_i)H(t_i) = -\frac{1}{l_d}\sum_{t_i\in d}\left(tf(t_i)\sum_{j=1}^{n_d}\frac{tf(d_j)}{tf(c)}\log\left(\frac{tf(d_j)}{tf(c)}\right)\right) \tag{4}$$

where $tf(t_i)$ is $t_i$'s term frequency in document $d$, and $l_d$ is the length of document $d$. The purpose of having a denominator $l_d$ here is to eliminate the influence of different document lengths.

We suppose that an important document should normally contain more discriminative terms (low entropy values) than a less important document (after normalization by the document length). Thus, the lower the entropy-based score, the more important the document is considered to be. Conversely, a less important document is expected to have a high entropy-based score. Our entropy-based pruning removes all the postings of less important documents from the index table according to the desired pruning rate.

## 4 Experimental Evaluation

All experiments were conducted using the LEMUR toolkit [5]. Documents were stemmed using the Krovetz stemmer [6]. However, note that stopwords were *not* removed during the pruning stage. Once the documents have been scored, and the required number of documents pruned from the collection, the remaining subset of the collection is indexed with stopwords removed. We use the stopword list suggested by Fox [7], which includes a total of 421 stopwords. In our evaluations, the "title" part and the "description" part of TREC topics are used as test queries. In all our experiments, the Okapi BM25 [8] scoring function was used. Note that the index has global parameters that are set based on the statistics of the collection, e.g. average document length, inverse document frequency, etc. Since our "collection" changes each time we prune it (remove documents), some performance variation may also be due to the changes of global index parameters.

Carmel's top-$k$ pruning algorithm is used as a baseline. The value of $k$ is set to 10 as in [1], and the different pruning levels are obtained by modifying the parameter $\epsilon$. We use the same measures as in [1], i.e. MAP and P@10, to evaluate the pruning. The percent of the index is defined as the ratio of the number of postings in the pruned index to that in the original index.

## 4.1 Comparison Based on FT and LA Collections

We compared the performance of our entropy-based method with that of Carmel's on the FT and LA collections. The FT collection consists of 210,158 documents, while the LA collection consists of 131,896 documents. For both collections, the TREC 6, 7 and 8 ad hoc topics (topics 301-450) are used to evaluate the pruning.

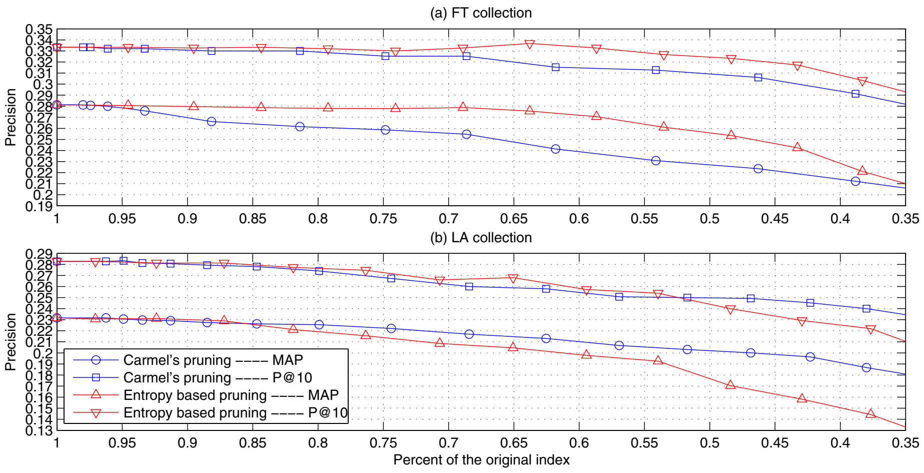

**Fig. 1.** Comparison of Carmel's pruning and entropy-based pruning on the FT and LA collections (evaluated by topics 301-450)

Figure 1 (a) is for the FT collection and shows that when the pruned index is 70% of its original size, there is little or no performance degradation for the entropy-based method. In contrast, for the same pruning level using Carmel's method, there is a 8.46% degradation in MAP and a 2.40% degradation in P@10. For all pruning rates, the entropy-based method exhibits improved performance compared with that of Carmel's.

Figure 1 (b) shows results for the LA collection. Here we observe that for 85% of the original index size, the entropy-based and Carmel's methods have similar performance for both P@10 and MAP. However, for further pruning, Carmel's methods is generally superior with respect to MAP.

### 4.2 Hybrid Method

The variation in performance across collections suggests that a hybrid approach might provide stability across collections. We therefore implemented a pruning method in which the first 10% of the index is pruned using the entropy-based method, and subsequent levels of pruning are provided by Carmel's method. Thus, for example, for an index with 30% pruning, one third of the pruning is due to the entropy-based method and two thirds is due to Carmel's method. We also considered other relative mixes of the two algorithms. However, space limitations preclude further discussion.

Figure 2 (a) compares the performance of the hybrid method with that of Carmel's for the FT collection, and Figure 2 (b) is the comparison for the LA collection. In both cases, the hybrid method performs as good as, or slightly better than Carmel's method. However, for the FT collection, we observe that the performance of the hybrid method is worse than our original entropy-based method. Further work is needed to develop a hybrid algorithm whose performance approaches that of the best individual algorithm.

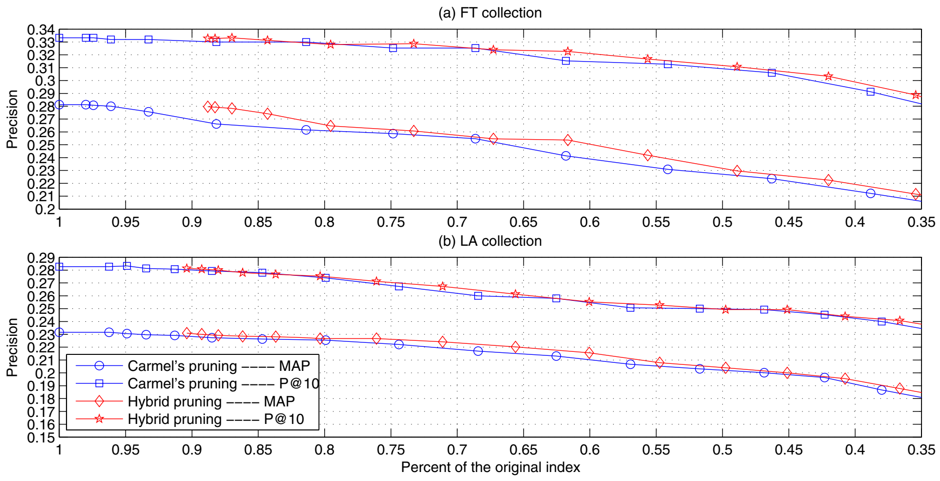

**Fig. 2.** Comparison of Carmel's pruning and the hybrid pruning strategy on the FT and LA collections (evaluated by topics 301-450)

## 5 Conclusions and Future Work

Previous static pruning algorithms have removed terms or postings from the index table. In this paper, we examined an third alternative, which is to remove documents from the collection. Documents are selected for pruning using an entropy-based score. Experimental results revealed that the entropy-based method was superior to Carmel's method on the FT collection, as measured using P@10 and MAP. However, Carmel's method was generally superior with MAP when tested using the LA collection.

The variation in performance across collections suggests that a hybrid method might offer some stability in performance across collections. The hybrid method first prunes up to 10% of the index using the entropy-based method. Subsequent pruning is then performed using Carmel's method. While a small improvement in performance is observed compared with Carmel's method for both test collections, the performance of the hybrid method is worse than the entropy-based method for the FT collection. Future investigation is needed to develop a hybrid algorithm in which the two original algorithms are combined in a more sophisticated manner.

The entropy-based method removes documents from the original index. Each pruned index therefore results in a slightly different set of global statistics, e.g. average document length. Since the retrieval scores are determined, in part, by the overall statistics of the index, a document present in the original and a pruned index may have a different retrieval score in response to the same query. This phenomenon was verified in experiments (not reported here) in which the global parameters were fixed to those derived from the original index, and used for all pruned indexes. In future work we would like to better understand this phenomenon.

We also intend to examine the Web Track of TREC (WT10G) in order to experiment with much larger collections. In addition, we would like to investigate whether our method can be adopted to provide a level of dynamic pruning.

## References

1. Carmel, D., Cohen, D., Fagin, R., Farchi, E., Herscovici, M., Maarek, Y.S., Soffer, A.: Static index pruning for information retrieval systems. SIGIR, 43–50 (2001)
2. Manning, C.D., Raghavan, P., Schtze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
3. Blanco, R., Barreiro, A.: Static pruning of terms in inverted files. In: Amati, G., Carpineto, C., Romano, G. (eds.) ECIR 2007. LNCS, vol. 4425, pp. 64–75. Springer, Heidelberg (2007)
4. Buttcher, S., Clarke, C.L.A.: A document-centric approach to static index pruning in text retrieval systems. In: CIKM, pp. 182–189 (2006)
5. Ogilvie, P., Callan, J.: Experiments using the lemur toolkit. In: Proceedings of the Tenth Text Retrieval Conference, TREC-10 (2001)
6. Krovetz, R.: Viewing morphology as an inference process. In: SIGIR, pp. 191–202 (1993)
7. Fox, C.: A stop list for general text. SIGIR Forum 24(1-2), 19–21 (1990)
8. Sparck-Jones, K., Walker, S., Robertson, S.E.: A probabilistic model of information retrieval. Information Processing and Management 36(6), 779–808 (2000)