Light Syntactically-Based Index Pruning for Information Retrieval

Christina Lioma and Iadh Ounis

University of Glasgow, G12 8QQ, UK
{xristina,ounis}@dcs.gla.ac.uk

Abstract. Most index pruning techniques eliminate terms from an index on the basis of the contribution of those terms to the content of the documents. We present a novel syntactically-based index pruning technique, which uses exclusively shallow syntactic evidence to decide upon which terms to prune. This type of evidence is document-independent, and is based on the assumption that, in a general collection of documents, there exists an approximately proportional relation between the frequency and content of 'blocks of parts of speech' (*POS blocks*) [5]. POS blocks are fixed-length sequences of nouns, verbs, and other parts of speech, extracted from a corpus. We remove from the index, terms that correspond to low-frequency POS blocks, using two different strategies: (i) considering that low-frequency POS blocks correspond to sequences of content-poor words, and (ii) considering that low-frequency POS blocks, which also contain 'non content-bearing parts of speech', such as prepositions for example, correspond to sequences of content-poor words. We experiment with two TREC test collections and two statistically different weighting models. Using full indices as our baseline, we show that syntactically-based index pruning overall enhances retrieval performance, in terms of both average and early precision, for light pruning levels, while also reducing the size of the index. Our novel low-cost technique performs at least similarly to other related work, even though it does not consider document-specific information, and as such it is more general.

1 Introduction

The field of Information Retrieval (IR) addresses the general problem of how to retrieve information, which is relevant to a user need, from a given repository of information, such as a document collection. Information in the document collection is represented in the form of an index, which contains statistics on term frequencies in each document and in the whole collection. An integral part of the index is the postings file, which records information on which terms appear in which documents and the term frequency statistics of these terms [7]. Usually, terms are associated with individual weights, which capture the importance of the terms to the content of each document. Term weights can be computed using various term weighting schemes. A matching function then estimates the likely

G. Amati, C. Carpineto, and G. Romano (Eds.): ECIR 2007, LNCS 4425, pp. 88–100, 2007.

relevance of a document to a query, on the basis of term weights, and the most relevant documents are identified and retrieved [11].

Very often, stopwords are removed from the index. Stopword removal is a way of pruning terms that harm retrieval performance. Generally, *index pruning* consists of replacing a full index by a smaller index, with the aim of improving system efficiency, without harming significantly retrieval performance [7]. System efficiency relates to issues such as the computational costs associated with storing large indices, or the time needed to query large indices. Retrieval performance relates to how relevant the returned results are (precision), or how many of the relevant results are returned (recall). Typically, system efficiency benefits from index pruning, because pruned indices tend to be more economical to store, and less time-consuming to query. Overall retrieval performance tends to decrease after index pruning, although early precision can be enhanced, for moderate or light pruning [4,10].

We present a light index pruning technique, which uses shallow syntactic evidence to prune low-content terms from the index, at indexing time. Specifically, this shallow syntactic evidence consists of blocks of part of speech (*POS blocks*), which are induced from a corpus. Any general corpus, such as a test collection, can be used. Firstly, a POS tagger maps every term in the corpus to a part of speech. We define a *POS block* as a fixed-length block of parts of speech, which we extract from text in a recurrent and overlapping way. For example, for a given sentence ABCDEFGH, where parts of speech are denoted by the single letters A, B, C, D, E, F, G, H, and where POS block length = 4, the POS blocks extracted are ABCD, BCDE, CDEF, DEFG, and EFGH. We extract all possible POS blocks from the corpus, without considering the order in which POS blocks occur, or the documents in which they occur. It has been shown that low-frequency POS blocks correspond to low-content words [5], unlike the case of individual words, where low-frequency words tend to be high in content [6]. On this basis, we hypothesise that pruning the words corresponding to low-frequency POS blocks from an index corresponds to eliminating content-poor words, and may enhance retrieval performance. In order to test the validity of this hypothesis, we take the following steps. Firstly, we POS tag a corpus and extract POS blocks from it. Secondly, we set a cutoff threshold θ, which controls which POS blocks from the corpus are used for pruning the index. Thirdly, we POS tag the collection to be indexed, and remove from it the words which correspond to POS blocks bounded by θ. With regards to which POS blocks from the corpus are used for index pruning, we test two different strategies, which we call *Rank A* and *Rank B*, respectively. *Rank A* considers the raw frequency of a POS block as indicative of the content salience of the words corresponding to that POS block. More simply, it assumes that low-frequency POS blocks correspond to low-content words. Using this strategy, POS blocks are frequency-sorted, and the θ least frequent POS blocks are used for index pruning. *Rank B* considers both the frequency of a POS block, and the part of speech classification (*POS class*) of its components, as indicative of the content salience of the words corresponding to that POS block. More simply, it assumes that low-frequency POS blocks that contain non content-bearing parts of speech, such as prepositions for example, correspond to low-content words. Using this strategy, POS blocks are sorted according to their frequency and member parts of speech, and the θ least frequent POS blocks are used for index pruning. Note that this is a low-cost approach, because it simply requires running the collection through the POS tagger once at indexing time, and is not born down by query or document-centric parameters.

This paper is organised as follows. Section 2 presents related work. Section 3 presents in details our proposed syntactically-based index pruning technique. Section 4 presents and discusses our experiments using *Rank A* and *Rank B* strategies. Section 5 summarises our findings and states intended future work.

2 Related Studies

Index pruning is used in IR to improve system efficiency, without harming significantly retrieval performance [4,7]. Typically, the data pruned from the index is estimated to be the least important to retrieval performance, according to some relevance criteria [2,4,7,10]. Index pruning is uniform when it is applied to all the documents in the same way, regardless of document- or term-specific criteria. A detailed overview of index pruning methods is given in [4]. In the same study, Carmel et al. investigate uniform and term-based index pruning methods, and report that early precision is not affected by moderate pruning, unlike average precision, which seems to decrease approximately linearly with the amount of data pruned. An alternative to pruning terms from an index, is replacing the documents in the index by their respective summaries [2,10]. Brandow et al. show that summary indexing improves precision at the cost of a large loss in recall [2]. This claim is also supported by Sakai and Sparck Jones, who report that moderate summary indexing does not affect early precision [10]. Overall, the consensus seems to be that light or moderate index pruning does not decrease significantly early precision, but may decrease average precision. As long as retrieval performance is not significantly hurt by index pruning, pruning techniques are applied, driven primarily by system efficiency gains [7].

The syntactically-based index pruning technique we present, differs from the above work in two ways. Firstly, our aim is not to mainly improve system efficiency, but also to enhance retrieval performance. Hence, we only and solely apply light pruning, wishing to validate solely the applicability of our novel syntactically-based pruning technique, and not its detailed effect upon general system efficiency, even though we do report on the index compression resulting from our pruning techniques, as compression in index is typically related to gains in efficiency. Secondly, the pruning criteria we use are not lexical, but exclusively shallow syntactic. More simply, our pruning criteria do not relate to words, but parts of speech. Also, our pruning technique does not use relevance weight metrics, or other document-specific criteria, to decide which terms to prune. The only two criteria used are the frequency of a POS block in a corpus, and the POS class of the members of a POS block, namely whether they are nouns, verb, prepositions, and so on. In this respect, our technique is novel, generic, and document-independent. Note that, in literature, restrictions are usually

Table 1. Open class parts of speech and their % frequency in WT10G

Part of Speech (POS)	POS Class	% in WT10G
Noun	open	38
Adjective		8
Verb		7
Participle		4

introduced in the pruning strategies. For example, in [4], terms are pruned only from the postings list, while in [10], there is a minimum length for a summary. On the opposite, our application includes no such restrictions. We prune terms from all the data structures of the index, and also allow for documents to have all of their terms pruned. Applying such restrictions may further refine our technique, and lead to further improving our reported results.

3 Syntactically-Based Index Pruning

We present the steps taken in order to test the hypothesis that shallow syntactic evidence can indicate low-content terms, whose elimination from the index can enhance retrieval performance, while reducing the index size.

All words in language are syntactically classified as either open or closed class words. The open class contains nouns, verbs, and generally content-rich words, while the closed class contains prepositions, conjunctions, and generally content-poor words that mainly perform linguistic well-formedness functions, instead of bearing content. These syntactic categories of words collectively constitute the *parts of speech*. Following from [5], we consider a shallow categorisation of parts of speech, namely one that only distinguishes between 14 parts of speech, as it is enough to distinguish between content words and stop words. Table 1 displays the 4 open class parts of speech we use, out of all 14. Using a POS tagger, we extract POS blocks from a corpus[1]. Section 1 presented and illustrated how POS blocks are extracted from text. At the end of this stage, we have a list of all the POS blocks induced from the corpus. In order to turn the list of POS blocks extracted from the corpus into evidence that can be used to indicate low-content terms, we test two different strategies, namely *Rank A* and *Rank B*.

Rank A. We consider the raw frequency of a POS block in the corpus as indicative of the content salience of the words potentially associated to that POS block [5]. We sort all the POS blocks extracted from the corpus in order of raw frequency, and assume that $low - frequency \approx low - content$.

Rank B. We consider both the frequency of a POS block, and the POS class of its components, as indicative of the content salience of the words corresponding to that POS block. The open class parts of speech we use are displayed in

[1] We use WT10G, but generally, any corpus can be used.

Table 1. In order to represent this combination of frequency and POS class in a quantitative way, we introduce an estimator of content for POS blocks. This estimator approximates the potential amount of content of the words corresponding to a POS block on the basis of: (i) the POS class of the POS block components, and (ii) the length of the POS block. This content score estimator is based on two assumptions, namely that: (i) only open class parts of speech correspond to content-bearing words (see Table 1); (ii) nouns are slightly more content-bearing than adjectives, verbs, and participles. Both of these assumptions are based on linguistic intuition. Specifically, in this paper, the content score $cs_{POSblock}$ of a POS block is estimated as follows:

$$cs_{POSblock} = \frac{C_N + C_{AVP} \cdot \varrho}{l_{POSblock}} \tag{1}$$

where C_N = number of nouns in the POS block, C_{AVP} = number of adjectives and/or verbs and/or participles in the POS block, $l_{POSblock}$ = POS block length, and ϱ = penalising parameter. ϱ is a penalising parameter applied to adjectives, verbs, and participles, following from the intuition that they are slightly less content-bearing than nouns. Using the statistics found in Table 1 as a guide, we set $\varrho = 0.17$, as follows. Adjectives, verbs and participles occur in the corpus approximately 19% (=8% + 7% + 4%) of the times, while nouns occur approximately 38% of the times. We estimate $\varrho = \frac{19/3}{38} \approx 0.17$. Using Equation (1), the content score of a POS block can be between 0 and 1, where 0 and 1 denote no content and the maximum amount of content, respectively. For example, using the here-proposed content score estimator for POS blocks, the POS blocks *noun + noun + noun + noun* and *adjective + noun + preposition + adverb* score $cs = 1$ and $cs = 0.29$, respectively.

Having established a quantitative estimator of content for POS blocks, we come back to the second strategy used to test our hypothesis. Specifically, we multiply the raw frequency and content score of POS blocks, and sort the POS blocks extracted from the corpus, according to the product of this multiplication. Multiplication is a simple way of combining frequency to content score, which practially implements our assumption, and hence is suitable for our experimentation. Other linear or log-scale combination approaches may be also used.

Strategies *Rank A* and *Rank B* are used for index pruning as follows. Firstly, identically to Section 1, we POS tag the collection to be indexed, only that this time we retain information on which POS blocks correspond to which terms in the collection. Secondly, we define a cutoff threshold θ to control the number of POS blocks used for index pruning. Then, only for those POS blocks bounded by θ, we remove from the collection the terms corresponding to those POS blocks. Terms are removed from all the data structures at indexing time. Note that the value of θ does not correspond to the actual frequency (for *Rank A*), or product of frequency and content score (for *Rank B*) of a POS block, but to the number of POS blocks which are to be used for index pruning. Note also, that, for the POS blocks selected for pruning to be low-content, we start counting θ POS blocks from the the lowest ranking to the higher ranking. For example, let us assume that *Rank A* contains 10 POS blocks, in decreasing order of frequency. Then, setting $\theta = 3$, means that the words corresponding to the 10nth, 9nth, and 8th POS blocks are pruned from the index.

4 Evaluation

4.1 Experimental Settings

We evaluate our hypothesis on WT2G (2GB) and WT10G (10GB), from the 1999, 2000 and 2001 Small Web, Web, and Adhoc tracks of the TREC Web Track, respectively, using topics 401-550 from the corresponding tasks[2]. We experiment with Title-only queries, as such queries are more representative of real user queries on the Web. During indexing, we apply stopword removal and Porter's full stemming. We select the largest of the two collections, namely WT10G, as the corpus from which we extract POS blocks, and POS tag it using the Tree-Tagger [3]. We set POS block length to $l_{POSblock} = 4$[4]. We the BM25 [9] and PL2 [1] weighting models. For all our experiments, we use the Terrier IR platform [8]. We use the default values of all weighting model parameters: (i) for BM25, $k1 = 1.2$, $k3 = 1000$, and $b = 0.75$ [9]; (ii) for PL2, $c = 10.99$ with WT2G, and $c = 13.13$ with WT10G[5]. We use default values, instead of tuning these parameters, because our focus is to test our index pruning hypothesis, and not to optimise retrieval performance. If the said parameters are optimised, retrieval performance may be further improved. We use mean average precision (MAP) and precision at 10 (P@10) to evaluate the impact of pruning on retrieval performance, using the full index as a baseline. We use a metric of similarity for the top k retrieved results, namely the *symmetric difference* [4] between the full and pruned indices, to evaluate the impact of pruning on early precision. We set k to 10, in accordance with P@10. The maximum and minimum symmetric difference scores of 1 and 0 occur when the top k results of the two indices are the same or disjoint, respectively, without considering the order of the results. In addition, we report the compression in index resulting from our pruning technique, as such compression is typically associated with gains in system efficiency.

4.2 Results and Discussion

We conduct experiments to test the hypothesis that pruning words which correspond to low-frequency POS blocks from the index, can enhance retrieval performance, at low pruning levels, using strategies *Rank A* and *Rank B*. By pruning levels, we denote the amount of data pruned from the full index.

[2] Information on the TREC datasets is found at: http://trec.nist.org/

[3] http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/

[4] We have also experimented with $l_{POSblock} = 3$ and 5, and observed little variation in retrieval performance. Generally, POS block length may vary, as long as it is kept within a range that appropriately models the adjacency of the terms in the sentence. This point is discussed in [5].

[5] Default settings for PL2 are suggested at:
http://ir.dcs.gla.ac.uk/terrier/doc/dfr_description.html

Table 2. Collection statistics after pruning. *POS blocks* θ = number of POS blocks used (in multiples of 1,000). *Rank A* = POS blocks sorted according to their frequency only. *Rank B* = POS blocks sorted according to the product of their frequency and content. *tokens* = individual words pruned (% from full index). *terms* = unique terms pruned (% from full index). *postings* = document pointers pruned from the postings list (% from full index).

POS blocks θ (1,000)	% pruned from full index						collection
	Rank A			**Rank B**			
	tokens	terms	postings	tokens	terms	postings	
23	18.39	5.07	14.56	20.37	14.09	15.42	WT2G
22	12.13	3.55	9.46	14.35	12.09	10.52	
21	8.47	2.16	6.53	10.99	10.78	7.77	
20	6.00	1.46	4.57	8.78	10.31	5.98	
19	4.38	1.10	3.31	7.36	9.97	4.82	
18	3.24	0.82	2.44	6.36	9.77	4.02	
17	2.44	0.65	1.83	5.70	9.61	3.49	
16	1.84	0.50	1.37	5.24	9.50	3.13	
15	1.41	0.39	1.04	4.90	9.42	2.86	
14	1.10	0.31	0.80	4.65	9.35	2.67	
13	0.82	0.24	0.61	4.46	9.30	2.52	
23	16.57	4.68	14.30	19.93	14.69	15.23	WT10G
22	11.16	3.23	9.38	14.57	12.72	10.56	
21	7.90	1.96	6.56	11.56	11.41	7.87	
20	5.70	1.39	4.65	9.62	11.00	6.12	
19	4.24	1.08	3.41	8.27	10.64	4.96	
18	3.33	0.84	2.55	7.37	10.47	4.17	
17	2.43	0.66	1.93	6.73	10.29	3.63	
16	1.86	0.53	1.46	6.28	10.17	3.26	
15	1.44	0.40	1.11	5.96	10.10	2.99	
14	1.13	0.31	0.86	5.71	10.04	2.78	
13	0.86	0.25	0.66	5.52	9.99	2.63	

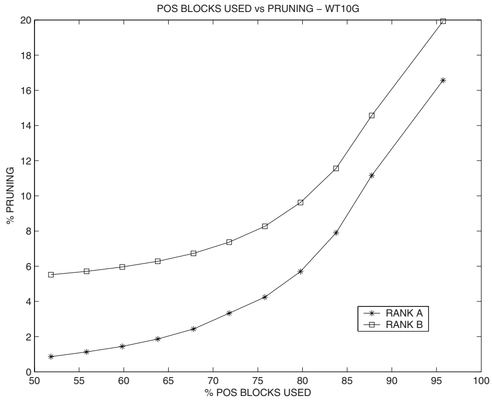

Fig. 1. % POS blocks used vs % pruning for WT10G

Table 2 displays statistics relating to the effect of each of our two pruning strategies on WT2G and WT10G, during indexing, separately for tokens, unique terms, and postings. Column *POS blocks θ (1,000)* contains the number of POS blocks used for pruning, in multiples of 1,000. We clearly see that pruning terms from more POS blocks (increasing θ) leads to more index compression (more terms being removed from the index). We also see that *Rank B* is more effective than *Rank A*, in the sense that it leads to more index compression, throughout. This is due to the fact that *Rank A* is a raw frequency sort of POS blocks, while *Rank B* is a sorting of a combination of POS block frequency and POS class information. More simply, the lowest ranked POS block in *Rank B* is not necessarily the least frequent POS block in the corpus, but a POS block that combines both (i) very low frequency, and (ii) very low content score. Very low content score practically translates to a POS block which does not contain any nouns, adjectives, or verbs. The fact that *Rank B* is better than *Rank A* is graphically illustrated in Figure 1, which plots various pruning levels against the % of POS blocks used. This % is the proportion of POS blocks used for pruning WT10G, out of all the POS blocks extracted from WT10G. We clearly observe that using the same number of POS blocks with *Rank A* and *Rank B* results in more index compression for the latter, than for the former. In Figure 1, we also observe that, even pruning the terms corresponding to 95% of all the POS blocks extracted from WT10G, only results in reducing the WT10G full index by 16-20%. Since we only use the lowest frequent POS blocks for pruning, this seems to indicate that there exists a very large number of POS blocks of low frequency, which is one of the properties of a power law distribution[6].

Table 3 displays retrieval performance scores at different index compression levels, separately for *Rank A* and *Rank A*, and for each collection. Pruning levels are reported in % reduction of postings, similarly to [4]. We see that light pruning leads to an overall improvement in MAP and P@10 over the full index, which is sometimes statistically significant. Two important observations are drawn from this table. Firstly, at no point does pruning hurt significantly retrieval. This point is very encouraging, considering that our techniques uses no document-specific criteria. Secondly, light pruning can improve both MAP and P@10. In fact, the best obtained MAP and P@10 scores for WT2G, namely MAP = 0.320 and P@10 = 0.468, are not given by the full index, but by pruning 2.86% and 1.37% of the index, respectively. Both of these scores are statistically very significant ($p << 0.01$). Similarly for WT10G, the best overall MAP and P@10 scores, namely MAP = 0.210 and P@10 = 0.328, are not given by the full index, but by pruning 0.66% and 2.99% of the index. The best overall retrieval scores are separately displayed in Table 4. Finally, in Table 3 we observe that PL2 performs better than BM25, which could be due to the default parameter settings used. Even so, both PL2 and BM25 outperform scores reported in [3,4], using TF·IDF and the same settings.

[6] Indeed we can report that the distribution of POS blocks in WT10G follows a Zipfian distribution.

Table 3. Pruning using POS blocks from Rank A and Rank B. *Prune* (%) = reduction in postings from full index. Grey-shaded = full index. Boldface = equal to or better than the full index. * and ** = stat. significance at $p < 0.05$ and $p < 0.01$ (Wilcoxon matched-pairs signed-ranks test), respectively.

Rank A					**Rank B**					
Prune (%)	MAP		P@10		Prune (%)	MAP		P@10		Collection
	BM25	PL2	BM25	PL2		BM25	PL2	BM25	PL2	
14.56	0.243**	0.298**	__0.432**__	0.454**	15.42	0.246*	0.301**	__0.430*__	0.454**	WT2G
9.46	0.250	0.303	__0.426__	__0.456__	10.52	0.254	0.308	__0.438__	__0.456__	
6.53	0.253	0.306	__0.428__	__0.466__	7.77	__0.258__	0.310	__0.432__	__0.466__	
4.57	__0.257__	0.310	__0.442__	__0.462__	5.98	0.257	0.311	__0.436__	__0.466__	
3.31	__0.259__	0.313	__0.442__	__0.460__	4.82	__0.259__	0.312	__0.434__	__0.464__	
2.44	__0.259__	0.313	__0.440__	__0.462__	4.02	__0.260__	__0.317__	__0.440__	__0.462__	
1.83	__0.260__	0.315	__0.442__	__0.464__	3.49	__0.259__	__0.319**__	__0.432__	__0.462**__	
1.37	__0.261*__	__0.317**__	__0.438*__	__0.468**__	3.13	__0.260__	__0.319**__	__0.434__	__0.460**__	
1.04	__0.261*__	__0.318*__	__0.438*__	__0.462*__	2.86	__0.258__	__0.320**__	__0.434__	__0.456**__	
0.80	__0.260__	__0.318__	__0.434__	__0.462__	2.67	__0.258__	__0.318__	__0.430__	0.454	
0.61	__0.260*__	__0.318*__	__0.436*__	__0.460*__	2.52	0.257*	__0.318__	__0.432*__	__0.456__	
0.00	0.258	0.317	0.426	0.456	0.00	0.258	0.317	0.426	0.456	
14.30	0.175**	0.195**	0.293**	0.298**	15.23	0.175**	0.199**	0.295**	0.307**	WT10G
9.38	0.182*	0.203	__0.304*__	0.307	10.56	0.179**	0.204	__0.300**__	0.307	
6.56	0.185*	0.206**	__0.302*__	0.316**	7.87	0.182*	0.207	__0.303*__	0.313	
4.65	__0.187__	0.207	__0.300__	0.317	6.12	0.184	0.207	__0.306__	0.316	
3.41	0.186	0.206	__0.301__	0.312	4.96	0.185	0.207	__0.302__	0.325	
2.55	__0.187__	0.208*	0.298	0.319*	4.17	0.185	0.208	__0.303__	0.325	
1.93	__0.187__	__0.209__	__0.300__	0.323	3.63	0.185	__0.209__	__0.301__	__0.326__	
1.46	0.186	__0.209__	__0.302__	0.324	3.26	0.186	0.208	__0.303__	__0.326__	
1.11	__0.187__	__0.209__	__0.302__	0.324	2.99	0.186*	__0.209__	__0.303*__	__0.328__	
0.86	__0.187__	__0.209__	__0.302__	__0.326__	2.78	0.186	__0.209__	__0.301__	__0.328__	
0.66	__0.188__	__0.210__	__0.303__	__0.326__	2.63	0.186*	__0.209*__	__0.301*__	__0.328*__	
0.00	0.187	0.209	0.300	0.326	0.00	0.187	0.209	0.300	0.326	

Figure 2 plots MAP and P@10 versus index pruning. For both WT2G and WT10G, and for index compression more than roughly 6% of the full index, the relation between average precision and pruning becomes practically decreasing linear, where as pruning increases, average precision decreases, for both *Rank A* and *Rank B*. For index compression less than 6%, varying pruning leads to variations in average precision, which can be either increasing or decreasing, but only slightly. Using *Rank A* results in more variations for this pruning range (<6%), than using *Rank B*, for both collections and with both weighting models. This seems to suggest that *Rank B* is more stable. With regards to precision at 10, index compression roughly less than 8-10% of the full index seems to generally increase precision, with the exception of using PL2 on the WT10G collection. For index compression more than 10% of the full index, there is a slight degradation in early precision. Overall, *Rank A* and *Rank B* perform very similarly, in terms of retrieval performance. BM25 and PL2 also behave very similarly, indicating that our conclusions drawn from these runs are consistent across two statistically different weighting models, hence they are general.

Figure 3 plots the similarity of the top 10 results to the full index versus pruning, using a symmetric difference [4] estimation. We observe that the early precision obtained by *Rank B* approximates the full index more closely than that obtained by *Rank A*. This observation, which is consistent for both collections

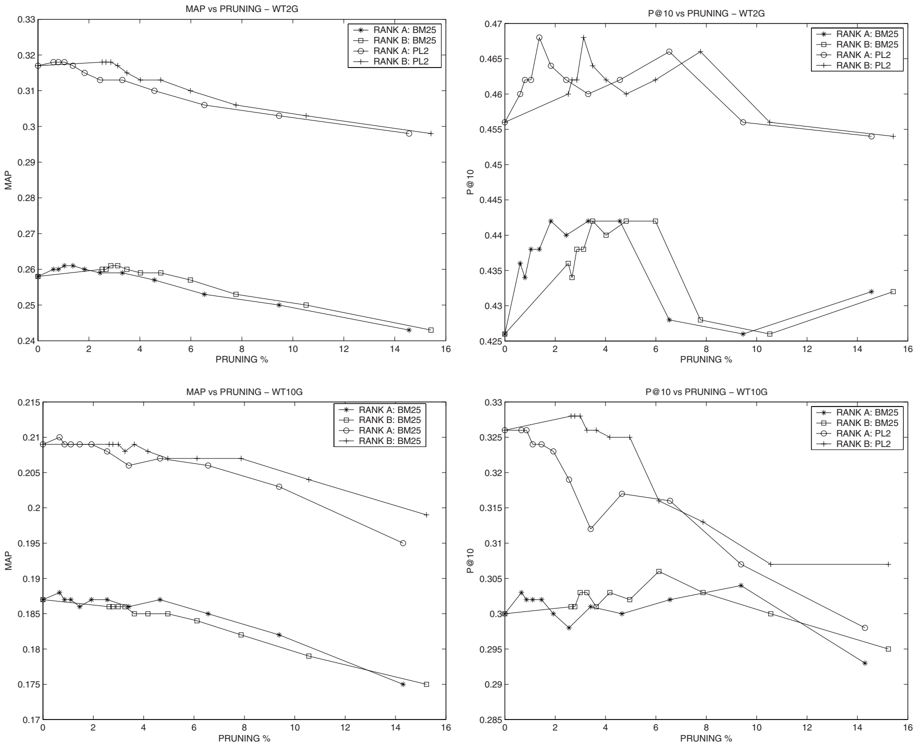

Fig. 2. Precision vs Pruning (% postings)

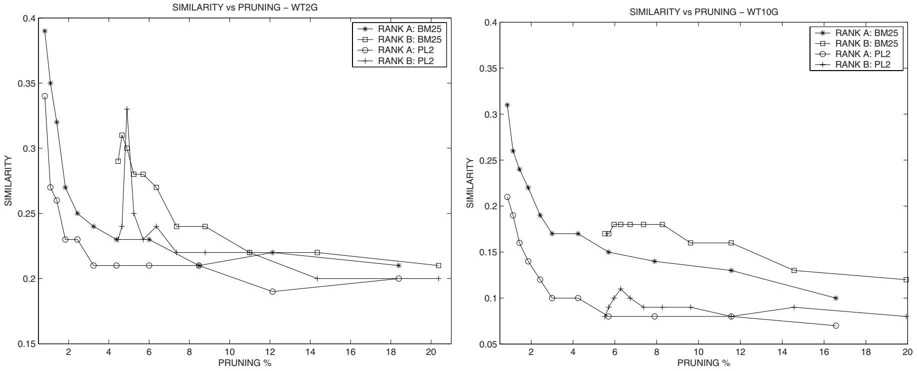

Fig. 3. Top 10 similarity at varying levels of pruning

and both weighting models, indicates that introducing the content score (Equation 1) of a POS block into the frequency ranking of POS blocks is a better pruning strategy.

Table 4. Best MAP and P@10 scores. $\Delta_F\%$ MAP & $\Delta_F\%$ $P@10$ = % difference from full index in MAP & P@10, respectively. *Prun.%* = % pruning in postings from full index. *strategy* = pruning strategy and weighting model. ** = stat. significance at $p < 0.01$ (Wilcoxon matched-pairs signed-ranks test).

best MAP	$\Delta_F\%$ MAP	Prun. %	strategy	best P@10	Δ_F % P@10	Prun. %	strategy	coll
0.320**	+1.0	2.86	Rank B: PL2	0.468**	+2.6	1.37	Rank A: PL2	WT2G
0.210	+0.5	0.66	Rank A: PL2	0.328	+0.6	2.99	Rank B: PL2	WT10G

Table 5 compares the performance of syntactically-based index pruning to other index pruning work [3,4]. We compare with reported experiments that use the same collections, topics, and similar pruning levels to ours (under column *experimental settings*). We see that, both *Rank A* and *Rank B* pruning strategies are at least comparable to the uniform pruning strategy of Carmel et al. [4], (marked ⊕). Note that we prune terms using document-independent syntactic evidence (Section 1), and from the whole index, while [4] prune terms according to their contribution to the relevance score of a document, and only from the postings lists. On the basis of these two key-differences, we consider the fact that our technique is comparable to that of [4], as very promising. Table 5 also includes the results of pruning reported in [3], (marked ⊗), whereby, in addition to using document-specific information and pruning the postings lists only, a term-based strategy is used. Our equivalent run applies a uniform, as opposed to term-based, technique, which is generally considered less effective [3,4]. Still, we observe no significant difference between the two runs, a fact which is an additional credit to our technique. When we repeat this run using the exact 50 topics used in [3], (marked ⋄), we observe that our technique outperforms that of [3] in P@10, with a slight decrease in MAP. We consider this performance notable, considering how much more refined is the pruning approach applied in [3], as already discussed.

Table 5. Comparison of our runs to other index pruning work (grey-shaded). $\Delta_F\%$ MAP & $\Delta_F\%$ $P@10$ = % difference in MAP & P@10 from full index. *Prun.%* = % pruning in postings from full index. ⊕ = run described in [4]. ⊗ = run described in [3]. ⋄ repeats the run of the preceeding row using 50, instead of 100, topics. Major differences appear in boldface.

Prun. %	$\Delta_F\%$ MAP	$\Delta_F\%$ $P@10$	experimental settings
15.4	-4.7	+0.9	WT2G, 401-450 (Title), Rank B, uniform prun. from all index
14.6	-5.8	+1.4	WT2G, 401-450 (Title), Rank A, uniform prun. from all index
⊕13.2	-4.0	+2.5	WT2G, 401-450 (Title), uniform prun. from **postings**
10.5	-1.6	+2.8	WT2G, 401-450 (Title), Rank B, uniform prun. from all index
⊗10.7	-1.9	0.0	WT10G, 501-550 (Title), **term-based** prun. from **postings**
10.6	-2.4	0.0	WT10G, 451-550 (Title), Rank B, uniform prun. from all index
⋄10.6	-2.9	+4.3	WT10G, 501-550 (Title), Rank B, uniform prun. from all index

5 Conclusion

We proposed a novel, low-cost, unsupervised statistical technique for index pruning, with uses shallow syntactic evidence to reduce noise from the index. We hypothesised that pruning the words corresponding to low-frequency POS blocks from an index corresponds to eliminating content-poor words, and may enhance retrieval performance. We presented POS blocks, as fixed-length blocks of parts of speech, and assumed that low-frequency POS blocks correspond to low-content words, following from [5]. On the basis of this, we tested two pruning strategies: Firstly, terms corresponding to θ low-frequency POS blocks, were pruned from the index (*Rank A*). Secondly, terms corresponding to θ low-frequency POS blocks which were also estimated to contain 'non content-bearing parts of speech', such as prepositions for example, were pruned from the index. We experimented with various values of θ, and reported on the associated effect on pruning levels and retrieval performance, while also making a note of the associated gain in index compression. Both strategies behaved similarly, with *Rank B* providing results closer to the full index, for early precision. Overall, by compressing the index up to a maximum of roughly 14-15% (see Table 3), our proposed syntactically-based pruned indices outperfomed the full indices, in terms of MAP and P@10, for both collections. Additionally, for similar index compression levels, our syntactically-based technique was shown to be comparable to [3,4], which used more refined document-specific and term-based pruning approaches. In the future, we wish to experiment with higher index compression levels, and also applying our pruning technique more intelligently, for example on a per-document basis.

References

1. Amati, G.: Probabilistic Models for Information Retrieval based on Divergence from Randomness. Phd thesis. Department of Computing Science, University of Glasgow (2003)
2. Brandow, R., Mitze, K., Rau, L.: Automatic Condensation of Electronic Publications by Sentence Selection. Information Processing and Management, 31(5). (1995) 675-685
3. Carmel, D., Amitay, E., Herscovici, M., Maarek, Y., Petruschka, Y., and Soffer, A.: Juru at TREC 10 - Experiments with Index Pruning. In: Text REtrieval Evaluation Conference (TREC 2001) 228-265
4. Carmel, D., Cohen, D., Fagin, R., Farchi, E., Herscovici, M., Maarek, Y., and Soffer, A.: Static Index Pruning for Information Retrieval Systems. In: ACM Conference on Research and Development in Information Retrieval (SIGIR 2001) 43-50
5. Lioma, C., Ounis, I.: Examining the Content Load of Part of Speech Blocks for Information Retrieval. In: Proceedings of the International Committee on Computational Linguistics and the Association for Computational Linguistics (COLING/ACL 2006)
6. Luhn, H., P.: The Automatic Creation of Literature Abstracts. (1958) 159-165
7. Witten, I. H., Moffat, A., Bell, T. C.: Managing Gigabytes: Compressing and Indexing Documents and Images. 2nd edn. Morgan Kaufmann, San Francisco (1999)

8. Ounis, I., Amati, G., Plachouras, V., He, B., Macdonald, C., Lioma, C.: Terrier: A High Performance and Scalable Information Retrieval Platform. In: ACM Conference on Research and Development in Information Retrieval Workshop on Open Source Information Retrieval (OSIR 2006)
9. Robertson, S., Walker, S.: Some Simple Approximations to the 2-Poisson Model for Probabilistic Weighted Retrieval. In: ACM Conference on Research and Development in Information Retrieval (SIGIR 1994) 232-241
10. Sakai, T., Sparck Jones, K.: Generic Summaries for Indexing in Information Retrieval. In: ACM Conference on Research and Development in Information Retrieval (SIGIR 2001) 190-198
11. van Rijsbergen, C., J.: Information Retrieval. Butterworths, London (1979)