Parsimonious Language Models for a Terabyte of Text

Djoerd Hiemstra[1] **Jaap Kamps**[2,3] **Rianne Kaptein**[3] **Rongmei Li**[1]

[1] Database Group, University of Twente
[2] ISLA, Informatics Institute, University of Amsterdam
[3] Archives and Information Studies, Faculty of Humanities, University of Amsterdam

Abstract:

The aims of this paper are twofold. Our first aim is to compare results of the earlier Terabyte tracks to the Million Query track. We submitted a number of runs using different document representations (such as full-text, title-fields, or incoming anchor-texts) to increase pool diversity. The initial results show broad agreement in system rankings over various measures on topic sets judged at both Terabyte and Million Query tracks, with runs using the full-text index giving superior results on all measures, but also some noteworthy upsets. Our second aim is to explore the use of parsimonious language models for retrieval on terabyte-scale collections. These models are smaller thus more efficient than the standard language models when used at indexing time, and they may also improve retrieval performance. We have conducted initial experiments using parsimonious models in combination with pseudo-relevance feedback, for both the Terabyte and Million Query track topic sets, and obtained promising initial results.

1 Introduction

The University of Amsterdam, in collaboration with the University of Twente, participated in the Million Query track with the twofold aim to compare results of the earlier Terabyte tracks to the Million Query track, and to explore the use of parsimonious language models for large-scale Web retrieval. When comparing results we specifically look at the impact of shallow pooling methods on the (apparent) effectiveness of retrieval techniques? And what is the impact of substantially larger numbers of topics?

We examine the use of parsimonious language models in combination with pseudo-relevance feedback. Since their introduction by Ponte and Croft [9] in 1998 statistical language models have become a major research area in information retrieval. The parsimonious language model overcomes some of the weaknesses of the standard language modeling approach where a mixture of the document model with a general collection model is used as follows:

$$P(t_1, ..., t_n|D) = \prod_{i=1}^{n} (\lambda P(t_i|D) + (1 - \lambda)P(t_i|C))$$

Instead of blindly modeling language use in a (relevant) document, we should model what language use distinguishes a document from other documents. The exclusion of words that are common in general English, and words that occur only occasionally in documents, can improve the performance of language models and decrease the size of the models. This so-called parsimonious model was introduced by Sparck-Jones et al. [11] and practically implemented by Hiemstra et al. [1].

The rest of this paper is organized as follows. We detail the experimental set-up for the tasks in the Terabyte track in Section 2. In Section 3, we discuss the details of the parsimonious language model. In Section 4, we discuss our official submissions and results as well as the results of our parsimonious language model experiments. Finally, we summarize our findings in Section 5.

2 Experimental Set-up

Our basic retrieval system is based on the Lucene engine with a number of home-grown extensions [2, 7].

2.1 Indexes

The Million Query track uses the `GOV2` test collection, containing 25,205,178 documents (426 Gb uncompressed). The indexing approach is similar to our earlier experiments in the TREC Web and Terabyte tracks [3, 4, 5, 6]. We created three separate indexes for

Full-text the full textual content of the documents (covering the whole collection);

Titles the text in the title tags of each document, if present (covering 86% of the collection);

Anchors another anchor-texts index in which we unfold all relative links (covering 49% of the collection).

For the anchor text index, we normalized the URLs, and did not index repeated occurrences of the same anchor-text. As to tokenization, we removed HTML-tags, punctuation marks, applied case-folding, and mapped marked characters into the unmarked tokens. We used the Snowball stemming algorithm [10]. The main full document text index was created as a single, non-distributed index. The size of our full-text index is 61 Gb. Building the full-text index (including all further processing) took a massive 15 days, 6 hours, and 21 minutes.

2.2 Basic Retrieval model

For our ranking, we use either a vector-space retrieval model or a language model. Our vector space model is the default similarity measure in Lucene [7], i.e., for a collection C, document D and query q:

$$sim(q,D) = \sum_{t \in q} \frac{tf_{t,q} \cdot idf_t}{norm_q} \cdot \frac{tf_{t,D} \cdot idf_t}{norm_D} \cdot coord_{q,D} \cdot weight_t,$$

where

$$\begin{aligned} tf_{t,X} &= \sqrt{\text{freq}(t,X)} \\ idf_t &= 1 + \log \frac{|C|}{\text{freq}(t,C)} \\ norm_q &= \sqrt{\sum_{t \in q} tf_{t,q} \cdot idf_t{}^2} \\ norm_D &= \sqrt{|D|} \\ coord_{q,D} &= \frac{|q \cap D|}{|q|} \end{aligned}$$

Our language model is an extension to Lucene [2], i.e., for a collection C, document D and query q:

$$P(D|q) = P(D) \cdot \prod_{t \in q} \left(\lambda P(t|D) + (1-\lambda) P(t|C)\right),$$

where

$$\begin{aligned} P(t|D) &= \frac{tf_{t,D}}{|D|} \\ P(t|C) &= \frac{\text{doc\_freq}(t,C)}{\sum_{t' \in C} \text{doc\_freq}(t',C)} \\ P(D) &= \frac{|D|}{\sum_{D' \in C} |D'|} \end{aligned}$$

The standard value for the smoothing parameter λ is 0.15. In previous years of the TREC Terabyte track, we found out that the GOV2 collection requires substantially less smoothing [3, 4]. That is, we use a value of λ close to 0.9.

2.3 Parsimonious Retrieval Model

Besides the official runs using the basic retrieval models, we also do a range of experiments with parsimonious language models. For efficiency reasons, we rerank in this case the top 1,000 results as produced by the official Lucene language model run UAmsT07MTeLM (using a full-text index, the Snowball stemming algorithm, standard multinomial language model with Jelinek-Mercer smoothing, $\lambda = 0.9$)

We compare a standard language model run using maximum likelihood estimation with a parsimonious retrieval model. A description of the parsimonious model follows below in a separate section. Pseudo-relevance feedback will be applied to both models. When a standard language model is used, we remove stopwords according to a standard stopwords list. To further improve performance we also apply Porter stemming in some of the runs.

3 The Parsimonious Language Model

The parsimonious model concentrates the probability mass on fewer terms than a standard language model. Only terms that occur relatively more frequent in the document as in the whole collection will be assigned a non-zero probability making the parsimonious language model smaller than a standard language model. The model automatically removes stopwords, and words that are mentioned occasionally in the document [1].

The model is estimated using *Expectation-Maximization*:

$$\text{E-step:} \quad e_t = tf(t,D) \cdot \frac{\lambda_p P(t|D)}{\lambda_p P(t|D) + (1-\lambda_p) P(t|C)}$$

$$\text{M-step:} \quad P(t|D) = \frac{e_t}{\sum_t e_t}, \text{i.e., normalize the model}$$

In the M-step the terms that receive a probability below a certain threshold or pruning factor are removed from the model. In the next iteration the probabilities of the remaining terms are again normalized. The iteration process stops after a fixed number of iterations or when the probability distribution does not change significantly anymore.

From a selection of health care pages from the GOV2 corpus, we built a standard and a parsimonious language model. In Table 1 the top ranked terms of both models are shown. The standard language model still contains some words that should be considered as stopwords, like 'shall'. When a standard stopword list is used there is always a trade-off between being complete and being too aggressive. When the parsimonious model is used, the document is compared to the background corpus to remove all words that do not occur relatively more frequently in the document than in the background corpus. In this way not only all standard stopwords are removed, but also the corpus specific stopwords. For example, in the GOV2 corpus the word 'information' can be considered a stopword because it occurs in almost half of all documents.

Table 1: Top ranked terms in the standard and the parsimonious language model.

Standard LM		Parsimonious LM	
Term	$P(t\|D)$	Term	$P(t\|D)$
cancer	0.0071	pharmacy	0.1186
pharmacy	0.0045	cancer	0.0673
board	0.0044	pharmacist	0.0650
health	0.0044	prostate	0.0460
shall	0.0040	diabetes	0.0336
care	0.0035	prescription	0.0332
patients	0.0033	patients	0.0213
research	0.0032	pharmacists	0.0174
drug	0.0030	ovarian	0.0158
state	0.0029	cancers	0.0143
treatment	0.0028	dispensing	0.0122
disease	0.0028	chemotherapy	0.0109
new	0.0027	prescriber	0.0104
information	0.0027	tumors	0.0099

Another advantage of the parsimonious model is that the probabilities are normalized. Frequently occurring words, such as stopwords, are removed from the model, and then the probabilities are redistributed over the remaining terms.

Pseudo-Relevance Feedback Since the parsimonious model concentrates on the most differentiating terms in a document, this model is a good candidate to use for pseudo-relevance feedback.

There are several possibilities for integrating pseudo-relevance feedback into the language model approach. We will use maximum likelihood ratios as described by Ng [8]. We calculate scores separately for the query and the relevance feedback likelihood ratio, and combine the scores at the end. The query and the feedback likelihood ratios differ substantially in length, therefore normalized log likelihood ratios should be used. For the query likelihood ratio, we then get:

$$NLLR_{query} = \log\left(\frac{P(Q|D,r)}{P(Q|D,\bar{r})}\right) = \sum_{i=1}^{n} P(q_i|Q) \cdot \log\left(\frac{\lambda P(q_i|D) + (1-\lambda)P(q_i|C)}{P(q_i|C)}\right)$$

To create the relevance feedback model $P(t_1, t_2, ..., t_n|R)$ we simply use the web-pages from the top scoring documents from our basic retrieval run. Here we use the top 10 results. The full-text of these web-pages is added together, and a (parsimonious) language model is created from this text. The relevance feedback model can be seen as an expanded weighted query. To estimate the feedback likelihood ratio, instead of summing over the query terms, we now sum over all terms contained in the feedback model as follows:

$$NLLR_{feedback} = \sum_{i=1}^{n} P(t_i|R) \cdot \log\left(\frac{\lambda P(t_i|D) + (1-\lambda)P(t_i|C)}{P(t_i|C)}\right)$$

The document model, i.e., $P(q_i|D)$ and $P(t_i|D)$, and the relevance feedback language model, $P(t_i|R)$, can be estimated according to the parsimonious model or according to the standard language model using maximum likelihood estimation. Since the two likelihood ratios are normalized, they can be easily combined. We have chosen the following formula to combine the query likelihood ratio and the relevance feedback likelihood ratio:

$$NLLR_{comb} = (1-\alpha) \cdot NLLR_{query} + \alpha \cdot NLLR_{feedback}$$

The α can depend on the quality of the initial run. If there P@10 is estimated to be low, the top 10 results are mostly not on topic, so it can be dangerous to use these results for pseudo-relevance feedback. In this case a smaller α should be used.

4 Experiments

We will first discuss the official runs, and analyse their results. Then, in a separate subsection, we discuss our initial experiments with parsimonious language models.

4.1 Official runs

We submitted five runs before, and three runs after the official deadline. Two further runs were used to construct the official submissions. Only the five official submissions have been part of the pooling process.

We submitted two runs on the full-text index run, using the vector space model (`UAmsT07MTeVS`) and using the language model (`UAmsT07MTeLM`, not pooled).

Next, we submitted a plain title index run (`UAmsT07MTiLM`) and a plain anchor-text index run (`UAmsT07MAnLM`) both using the language model. We also have the similar runs using vector-space model, using the title index (`UAmsT07MTiVS`, not submitted) and the anchor-text index (`UAmsT07MAnVS`, not submitted).

These separate indexes can provide additional retrieval cues, for example, the anchor-texts provide a document representation completely disjoint from the document's text. Hence, we also submitted four runs that combine different sources of evidence. First, a weighted CombSUM with relative weights of 0.6 (text), 0.2 (anchors), and 0.2 (titles) using the vector space model (`UAmsT07MSum6`) and the language model (`UAmsT07MSm6L`, not pooled). Second, a similar combination with relative weights of 0.8 (text), 0.1 (anchors), and 0.1 (titles), again using using the vector space model (`UAmsT07MSum8`) and the language model (`UAmsT07MSm8L`, not pooled).

Table 2: Statistics over judged and relevant documents per topic for million query track (top) and terabyte tracks (bottom).

	nr. of	per topic				
	topics	min	max	median	mean	st.dev
judged	1,778	6	147	40	41.07	6.58
relevant	1,524	1	52	10	12.23	9.62
high. rel.	745	1	44	3	5.36	6.17
judged	149	317	1,876	870	908.40	342.44
relevant	149	4	617	130	180.65	149.16
high. rel.	125	1	331	14	34.81	51.95

Table 3: Results for the MQ track.

	Million Query		Terabyte 2004-2006		
UAmsT07	NEU	UMass	map	bpref	P@10
...MTeVS	0.1805	0.0500	0.1654	0.2527	0.3047
...MTeLM	0.2908	–	0.2921	0.3410	0.5376
...MTiVS	0.0884	–	0.0369	0.0939	0.2168
...MTiLM	0.0938	0.0281	0.0392	0.0977	0.2154
...MAnVS	0.0561	–	0.0274	0.0763	0.2081
...MAnLM	0.0650	0.0205	0.0278	0.0742	0.2034
...MSum6	0.1816	0.0557	0.1398	0.2348	0.2953
...MSm6L	0.2255	–	0.2347	0.3069	0.3738
...MSum8	0.1995	0.0579	0.1621	0.2482	0.3094
...MSm8L	0.2867	–	0.2696	0.3273	0.4711
Topics	1,153	1,778	149	149	149

4.2 Official Run Results

The topic set contains 10,000 topics numbered 1 to 10000. Table 2 (top half) shows statistics of the number of judged and relevant documents, based on the final "prels" files. In total 1,778 different topics have been assessed. The number of relevant documents per topic varies from 1 to 52, with a mean of 12 and a median of 10. For no less than 253 topics, no relevant document has been found. The topic set also includes the adhoc topics of the Terabyte (TB) tracks at TREC 2004-2006. For comparison, we also show their statistics in Table 2 (bottom half). During the three years of the Terabyte track 149 topics have been assessed, with 4 to 617 relevant documents (mean 181 and median 130). There are striking differences between the two sets of judgments: First, the number of topics assessed at the MQ track is roughly ten times larger than the three year of TB track together. Second, the number of judged documents, as well as the number of relevant documents per topic is over ten times larger for the TB topics.

Table 3 shows the results for the MQ track. The first two scores are based on the MQ judgments: NEU stands for the estimated MAP (statMAP) as produced by the Northeastern University's method, UMass stands for the expected MAP as produced by the University of Massachusetts Amherst's method.[1] Comparing the scores over the five runs, we see that they are in complete agreement about the ranking. Both NEU and UMass methods agree on the best of the five runs: the vector-space combination (Sum8). Over all runs, the NEU method gives the highest statMAP score to the full-text language model run (TeLM). The next three scores in Table 3 are based on the TB assessments. The best scoring run on all measures is the full-text language model run (TeLM). The order of the five official submissions is different: now the full-text vector-space run (TeVS) scores best on MAP.

Table 4: Results for the MQ track using the shallow judgments as qrels.

	Million Query		
UAmsT07	map	bpref	P@10
...MTeVS	0.1684	0.2991	0.1644
...MTeLM	0.2818	0.3987	0.2703
...MTiVS	0.0841	0.1839	0.1108
...MTiLM	0.0924	0.1789	0.1211
...MAnVS	0.0604	0.1438	0.0940
...MAnLM	0.0695	0.1408	0.1072
...MSum6	0.1759	0.3006	0.1844
...MSm6L	0.2164	0.3663	0.2089
...MSum8	0.1905	0.3052	0.1899
...MSm8L	0.2788	0.4006	0.2638
Topics	1,524	1,524	1,524

What if we treat the MQ judgments as as normal qrels (so assuming for most measures that non-judged documents are non-relevant)? Table 4 shows the results. The best scoring run, again on all measures, is the full-text language model run (TeLM). The best official submission is the vector-space combination (Sum8), in agreement with both the NEU and UMass methods. In fact, the five official submission get the same order by MAP and by the NEU and UMass methods. More generally, we see that map and precision at 10 are resulting in the same system ranking, and that the precision at 10 scores are much lower than for the TB topics in Table 3. This is a clear indication that we have only unearthed a small sample of the relevant documents.

We have now shown three "qrels" and eight measures, how do these agree? Table 5 shows Kendall's tau of the system rank correlation. Some observations present themselves: First, we see that there is reasonable correlation between all pairs of measures, with correlations ranging from 0.6 to 1.0, with the 0.6 for the agreement between UMass and TB map, and UMass and TB bpref. Second, the agreement between NEU and Terabyte MAP (0.911 over 10 systems) seems higher than that of UMass (0.600 over 5 systems), however this may be misleading since the NEU measure ranks the 5 official runs in the exact same order as UMass.

[1] We failed to reproduce the "official" scores, and hence only include these for the five official runs.

Table 5: Rank correlations of the resulting system rankings (columns and rows are in the same order).

Million Query			Terabyte		Million Query		
NEU	UMass*	map	bpref	P@10	map	bpref	P@10
–	1.000	0.911	0.867	0.867	1.000	0.867	1.000
–	–	0.600	0.600	0.800	1.000	1.000	1.000
–	–	–	0.956	0.867	0.911	0.778	0.911
–	–	–	–	0.911	0.867	0.822	0.867
–	–	–	–	–	0.867	0.911	0.867
–	–	–	–	–	–	0.867	1.000
–	–	–	–	–	–	–	0.867
–	–	–	–	–	–	–	–

* Comparisons are restricted to 5 runs.

Table 6: Relevant, nonrelevant, and unjudged documents for MQ judged topics (top) and TB judged topics (bottom).

	Rank	Relevant		Nonrelevant		Unjudged	
		#	%	#	%	#	%
Text	1	591	33.24	719	40.44	468	26.32
	10	4,112	23.13	5,927	33.34	7,672	43.15
	100	12,410	6.98	22,596	12.71	141,628	79.66
	1,000	16,655	0.94	36,740	2.07	1,702,145	95.73
Anchors*	1	380	21.37	944	53.09	454	25.53
	10	1,645	9.25	4,977	27.99	10,950	61.59
	100	3,162	1.78	9,104	5.12	161,777	90.99
	1,000	4,475	0.25	11,886	0.67	1,660,685	93.40
Text*	1	82	55.03	67	44.97	0	0.00
	10	801	53.76	682	45.77	7	0.47
	100	5,726	38.43	8,039	53.95	1,135	7.62
	1,000	17,840	11.97	38,754	26.01	92,406	62.02
Anchors	1	52	34.90	56	37.58	41	27.52
	10	302	20.27	545	36.58	643	43.15
	100	1,319	8.85	3,821	25.64	9,643	64.72
	1,000	2,849	1.91	11,498	7.72	127,945	85.87

* Run was in the pool.

What is the impact of low pooling depth? We look at the number of relevant, nonrelevant, and unjudged documents in runs both inside and outside of the judgment pools. The results are shown in Table 6. Looking at the 1,778 MQ topics, over 25% of the top 1 results have not been judged. At rank 10, the percentage of unjudged documents is 43% (full-text, not pooled) and 62% (anchor-texts, pooled). The relative precision over judged documents is still 41% (full-text) and 25% (anchor-texts) suggesting strongly that the judgments are merely a sample. A clear call for caution to use the MQ judgments as traditional qrels (as we did in Table 4). For the MQ topics we see no significant difference between the coverages of runs in and outside the pools. In a sense this may make the comparison of official and post-submission runs less unfair. Looking at the 149 TB topics, we see clearly the difference in the percentage of judged documents for the pooled run (full-text, very similar runs were in the top 50 pools at TREC 2004-2006), and outside the pool (anchor-texts).

Table 7: Results of the parsimonious models on the Terabyte topics for non-stemmed (top half) and stemmed (bottom half) documents.

Model	λ_p	map	bpref	P@10
MLE		0.2189	0.2985	0.3463
Parsimonious	0.01	0.2199	0.2994	0.3926
Parsimonious	0.1	0.2311	0.3125	0.3866
Parsimonious	0.2	0.2295	0.3104	0.3805
MLE (stemmed)		0.2183	0.3041	0.3342
Pars. (stemmed)	0.1	0.2359	0.3190	0.4040

4.3 Parsimonious Language Model Runs

We have not implemented the parsimonious language model in Lucene, but instead use our own scripts for a standard language model, and for the parsimonious language model. As a result, these runs have some limitations that affect the performance (as we will see below). For efficiency reasons, the background corpus does not consist of the whole GOV2 corpus, but of a random 1% sample of the GOV2 corpus. Another difference with the Lucene language model run is that we do not apply document length normalization in our runs. Moreover, since we rerank only the top 1,000 documents of each topic, possibly relevant documents outside these top 1,000 results are not being considered. Finally, our scripts used a more simple tokenization and stemming than was used in the original runs.

Before applying our models on the topics of the Million Query track, we first apply them on the 149 Terabyte track topics. As mentioned before, in previous years of the TREC Terabyte track we already noticed the GOV2 collection requires little smoothing. Initial experiments show in our setting even less smoothing is needed than in the Lucene language model runs, i.e. we set $\lambda = 0.99$. For the parsimonious model we also have to set the parameter λ_p^D. In previous research [1] values from 0.01 to 0.2 for λ_p lead to optimum performance depending on the exact task. We take the results of the query likelihood ratio as a baseline, and optimize on this baseline the parameter λ_p^D for our document model. Also we examine the influence of stemming on the baseline run. For the relevance feedback model we will optimize the parameter λ_p^R separately.

4.4 Parsimonious Model Results

The baseline provides us with a first indicator of the quality of the parsimonious model without the influence of the pseudo-relevance feedback. Table 7 (top half) shows the results for the non-stemmed runs. First we see that the score of the reranking are somewhat lower than for the official runs, because of the differences listed above. However, we also

see that the parsimonious models score better on all three evaluation measures than the standard language model with maximum likelihood estimation. With $\lambda_p^D = 0.1$ or 0.2, MAP and Bpref improve significantly (bootstrap test, one-tailed, $p < 0.05$) with around 5% and P@10 improves significantly with almost 12%. These initial experiments further show we get the best results with $\lambda_p^D = 0.1$. Although P@10 is better for $\lambda_p^D = 0.01$, MAP and Bpref are significantly worse. The results for $\lambda_p^D = 0.1$ are somewhat better than for $\lambda_p^D = 0.2$, but the difference is not significant. In the next experiments we use $\lambda_p^D = 0.1$ for the document model.

When we apply Porter stemming to the best baseline runs, shown in the bottom half of Table 7, the results do not improve much or not at all depending on the evaluation measure and the model. The standard language model with stemming gets a lower MAP and P@10, but a slightly higher Bpref. The parsimonious model achieves the largest improvement on P@10; MAP and Bpref only improve slightly. Considering that our adhoc implementation is faster when stemming is not used, and stemming does not result in a clear improvement on the baseline run, we will not apply stemming to the subsequent runs.

In the next experiment we again run the 149 Terabyte topics, and this time we do apply the pseudo-relevance feedback. The experiment is run with values for α ranging from 0, only the query likelihood ratio, to 1, only the relevance feedback ratio, with steps of 0.1. The parameter λ_p has a different optimal value for the relevance feedback model than for the document model. Experiments show that λ_p^R is best set at 0.01 for the relevance feedback model.

The optimal value for α depends on the evaluation measure that you want to optimize, MAP, Bpref or P@10. The parsimonious language model produces the best results for all three evaluation measures, however the measures are optimal at three different values for α. MAP peaks at $\alpha = 0.5$ with a value of 0.2953; Bpref at $\alpha = 0.6$ with a value of 0.3716; and P@10 is optimal at $\alpha = 0.4$ with a value of 0.5530. The MAP scores of all three models on the different values for α are plotted in Figure 1. When we compare the best run of the standard language model with the best run of the parsimonious model (both with $\alpha = 0.5$), Map and Bpref of the parsimonious model run are significantly better, with improvements of 5.2% and 7.3% respectively (see Table 8). P@10 is slightly better, improving 2.3%, but the difference is not significant.

We also experiment with a mixed model using maximum likelihood estimation for the document model and parsimonious estimation for the pseudo-relevance feedback model. The scores are also shown in Figure 1, and are in between the two other runs, i.e better than the standard language model, and not as good as the parsimonious model.

We have also tested our models on the Million Query topics (using the initial set of 1,692 topics). The results

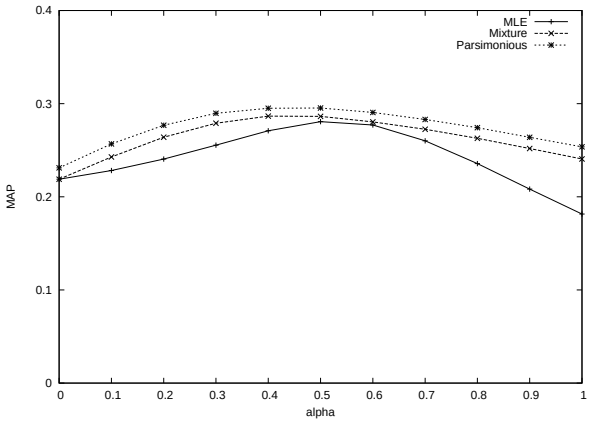

Figure 1: Blind feedback results on the Terabyte topics (MAP).

Table 8: Results of the parsimonious models on the Terabyte and MQ topics (using the initial set of 1,692 topics).

	Terabyte 2004-2006			MQ track
Model	map	bpref	P@10	NEU
MLE	0.2807	0.3458	0.5349	0.1824
Parsimonious	0.2953	0.3711	0.5470	0.1850

of the best runs can be found in Table 8 and are achieved with $\alpha = 0.4$ for the standard model, and $\alpha = 0.2$ for the parsimonious model. The NEU statMAP scores are lagging far behind the original score (`UAmsT07MTeLM` scores a statMAP of 0.2986 on these 1,692 topics). The limitations seem to affect the MQ topics (selected from a search engine's query log) far more severe than the Terabyte topics. For example, the sample used as collection model does not contain all query terms. Despite these limitations, we see again that the parsimonious language is superior to the standard MLE model. When we treat the MQ judgments as as normal qrels, the results are consistent with the NEU statMAP scores, i.e., the parsimonious run is better than the standard language model run.

4.5 Results Analysis

When we compare the results of the official runs with our runs with the standard and the parsimonious language model, the unofficial runs, we have to take into account that we are reranking the results of the official run. Stemming for example does not improve the unofficial runs much, possibly because stemming is already applied in the basic retrieval run, so documents with different terms but with the same stem are retrieved anyway.

However, despite the limitations mentioned in the beginning of the previous section, the best unofficial run (using parsimonious language models and $\alpha = 0.5$) on the Terabyte topics is better than the best official run, `UAmsT07MTeLM` that is also used as the run to rerank (see Table 8). While

the improvement in MAP and P@10 is small, under the 2%, there is a significant improvement in Bpref of 8.8%. In the unofficial runs it is the application of pseudo-relevance feedback that leads to the biggest improvement. In the official runs no feedback is applied.

The performance of our standard and parsimonious model on the Million Query topic is of much poorer quality than the performance on the Terabyte topic set. The topic set is the major difference between the Million Query runs and the Terabyte runs, along with the less complete judgments. The Million Query topics are more specific and contain quite some words that are not contained in our 1% sample of the background corpus. This is a possible explanation for the poorer results. Furthermore, since P@10 for the Million Query topics is roughly half of that of the Terabyte topics, 0.2703 and 0.5376 respectively, in combination with the much lower number of relevant documents, the quality of our pseudo-relevance feedback is lower, which is reflected in the lower values of α that give the best results. We are currently implementing our models in a standard search engine, which will overcome the current limitations such as the use of a sample as collection model, to further investigate the utility of parsimonious language models for Web retrieval.

5 Conclusions

During the TREC 2007 Million Query track, we submitted a number of runs using different document representations (such as full-text, title-fields, or incoming anchor-texts), and compared results of the earlier Terabyte tracks to the Million Query track. The initial results show broad agreement in system rankings over various measures on topic sets judged at both Terabyte and Million Query tracks, with runs using the full-text index giving superior results on all measures, but also some noteworthy upsets.

We also conducted initial experiments with parsimonious language models. We found that the parsimonious language model is to be preferred over the standard language model using maximum likelihood estimation. It leads to superior retrieval results, while at the same time using smaller document models (and hence reducing the index) and obliviate the need for stopword lists.

Acknowledgments

This research is part of the *Effective Focused Retrieval Techniques* (EfFoRT) project, funding by the Netherlands Organization for Scientific Research (NWO, grant # 612.066.513). We further acknowledge support by NWO (grants # 639.072.601, and 640.001.501), and by the E.U.'s 6th FP for RTD (project MultiMATCH contract IST-033104).

References

[1] D. Hiemstra, S. Robertson, and H. Zaragoza. Parsimonious language models for information retrieval. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 178–185. ACM Press, New York NY, 2004.

[2] ILPS. The ILPS extension of the Lucene search engine, 2008. http://ilps.science.uva.nl/Resources/.

[3] J. Kamps. Effective smoothing for a terabyte of text. In *The Fourteenth Text REtrieval Conference (TREC 2005)*. National Institute of Standards and Technology. NIST Special Publication, 2006.

[4] J. Kamps. Experiments with document and query representations for a terabyte of text. In *The Fifteenth Text REtrieval Conference (TREC 2006)*. National Institute of Standards and Technology. NIST Special Publication, 2007.

[5] J. Kamps, C. Monz, M. de Rijke, and B. Sigurbjörnsson. Approaches to robust and web retrieval. In *The Twelfth Text REtrieval Conference (TREC 2003)*, pages 594–599. National Institute of Standards and Technology. NIST Special Publication 500-255, 2004.

[6] J. Kamps, G. Mishne, and M. de Rijke. Language models for searching in Web corpora. In *The Thirteenth Text REtrieval Conference (TREC 2004)*. National Institute of Standards and Technology. NIST Special Publication 500-261, 2005.

[7] Lucene. The Lucene search engine, 2008. http://lucene.apache.org/.

[8] K. Ng. A maximum likelihood ratio information retrieval model. In *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*, 1999.

[9] J. Ponte and W. Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st ACM Conference on Research and Development in Information Retrieval (SIGIR'98)*, pages 275–281, 1998.

[10] Snowball. Stemming algorithms for use in information retrieval, 2008. http://www.snowball.tartarus.org/.

[11] K. Sparck-Jones, S. Robertson, D. Hiemstra, and H. Zaragoza. Language modelling and relevance. In W. Croft and J. Lafferty, editors, *Language Modeling for Information Retrieval*, pages 57–71. Kluwer Academic Publishers, 2003.