Experiments with ClueWeb09: Relevance Feedback and Web Tracks

Mark D. Smucker[1], Charles L. A. Clarke[2], and Gordon V. Cormack[2]

[1]Department of Management Sciences, University of Waterloo
[2]David R. Cheriton School of Computer Science, University of Waterloo

Abstract

In this paper, we report on our TREC experiments with the ClueWeb09 document collection. We participated in the relevance feedback and web tracks. While our phase 1 relevance feedback run's performance was good, our other relevance feedback and web track submissions' performances were lacking. We suspect this performance difference is caused by the Category B document subset of the ClueWeb09 collection having a higher prior probability of relevance than the rest of the collection. Future work will involve a more detailed error analysis of our experiments.

1 Introduction

Our goals for TREC this year were to get experience with the multi-terabyte ClueWeb09 document collection, and to establish baselines on the collection with various retrieval techniques. As such, we participated in the relevance feedback (RF) track as well as both the ad-hoc and diversity tasks of the Web track. Each of these tracks utilized the same 50 topics and document collection. We first briefly describe how we indexed ClueWeb09 and then explain our approaches for each of these tracks.

2 Indexing ClueWeb09 English

To index ClueWeb09 English (ClueWeb09_English_[1-10]), we used Indri [6] version 4.10, which had been modified by the Lemur Project developers to index ClueWeb09. We built 30 separate indexes across 10 machines. Each machine was a modern (2-2.3 GHz CPU) 4 core machine with 8 GB of RAM and approximately 800 GB of disk storage. Building the indexes took less than 2 days using 3 index processes per machine. The CatB subset (English_1) took 31.5 hours to index. Overall time spent to index the collection was about 2 weeks of work including moving 2 TB of data across the Internet. We stemmed all terms with the Krovetz [4] stemmer and did not perform stop word removal at index time.

3 Relevance Feedback Track

The relevance feedback track was run in two phases. The first phase allowed participating sites to submit two runs. Each run was limited to have at most 5 results for each topic and participants could only use the CatB subset of the ClueWeb09 collection. NIST had the assessors completely judge the Phase 1 runs for relevance. Phase 2 of the RF track consisted of submitting a baseline (no feedback) run of 2500 results and then runs utilizing the judgments from the phase 1 runs. Each site was assigned its own phase 1 runs as well as 5 other sites' phase 1 runs to use for feedback.

3.1 Phase 1 Methods

For our phase 1 run, we followed the technique of Metzler et al. [8] that combines dependence models [7] and relevance models [5]. Table 1 shows our retrieval parameter settings. We call this formulation DMRM3 to represent a dependence model (DM) mixed with a relevance model (RM). The successful technique of mixing a relevance model with the original query has come to be called RM3 [1].

In this formulation, the relevance model is typically computed from the top documents retrieved by the dependence model. For our phase 1 run, we obtained the pseudo-relevant feedback documents from the Yahoo (`search.yahoo.com`) and Bing (`www.bing.com`) web search engines. Our reasoning for this was to obtain from the smaller CatB collection a retrieval that would mimic as much as possible the assumed good retrievals of the major search engines.

For each of the topics, we retrieved the top 10 results from both Yahoo and Bing using the various APIs provided by the search engines. We also ob-

Parameter	Value
Dirichlet smoothing for unigram terms, m	1500
Dirichlet smoothing for ordered and unordered windows, m	4000
Weight of unigram model in dependence model	0.8
Weight of ordered windows model in dependence model	0.1
Weight of unordered windows model in dependence model	0.1
Weight of dependence model when mixed with pseudo relevance model	0.3
Weight of pseudo relevance model when mixed with dependence model	0.7
Maximum number of terms in pseudo feedback relevance model (Phase 1 web expansion and ad-hoc web track run WatSdmrm3we)	50
Maximum number of terms in pseudo feedback relevance model (Phase 2 RF and ad-hoc web track run WatSdmrm3)	25
Number of pseudo-relevant feedback documents (Phase 2 RF and ad-hoc web track run WatSdmrm3)	10

Table 1: Retrieval parameters.

tained results from Google, but were unable to utilize them because of technical issues and a lack of time.

For each result, we fetched the actual web page or pdf document. To easily compute a model of the documents, for each topic, we created an Indri index of the, up to, 20 documents. We stopped the collection with an in-house list of 418 stop words (see Appendix A of [9]) and we stemmed with the Krovetz stemmer [4]. After building each index, we computed a maximum likelihood estimated (MLE) collection model for each collection. In effect, we concatenated the documents and built a MLE language model. We truncated each model to the 50 terms with the highest probability. We then mixed this model with the dependence model.

We made a mistake in the building of the final DMRM3 queries. The web expansion component was weighted as part of the original query instead of mixing the DM query with the RM query. Instead of the queries being constructed as #weight(0.3 dependence_model 0.7 pseudo_relevance_model), we constructed them as #weight(0.3 #weight(0.8 unigram_model 0.1 ordered_windows 0.1 unordered_windows 0.7 pseudo_relevance_model)), which is the same effectively as #weight(0.8 unigram_model 0.1 ordered_windows 0.1 unordered_windows 0.7 pseudo_relevance_model).

We retrieved the top 10 documents using these queries. We did stop word removal at query time with the above described 418 stop words. We call this run WatS.1+WatS.2.

We created two phase 1 runs from WatS.1+WatS.2. WatS.1 consisted of the rank 1-5 results and WatS.2 consisted of ranks 6-10. By doing this we were able to have this CatB run judged to a depth of 10.

3.2 Phase 2 Methods

Our phase 1 runs, like any other runs utilizing commercial web search engines, utilize the secret retrieval methods of the web search engines. As such, our phase 1 runs are non-reproducible. While it is interesting to see what is possible with such expansions, these expansions are not the equivalent of doing one's own expansion on a web sized collection.

For our phase 2 baseline, WatS.base, we again utilized a dependence model with a relevance model for pseudo relevance feedback, but this time the initial retrieval was done with the dependence model. We built the relevance model from the top 25 results. The documents in the baseline pseudo-relevance model are weighted based on the dependence model's retrieval scores as is standard for relevance models.

Our relevance feedback runs used the following phase 1 runs: fub.1, SIEL.1, Sab.1, UCSC.1, twen.1, twen.2, WatS.1, and WatS.2. If the given phase 1 run had any relevant documents, we built a model of those relevant documents and replaced the pseudo-relevance model, otherwise the baseline query was used without change. We weighted the documents in the relevance model equally.

For our baseline and feedback runs, we computed a non-traditional relevance model. In our early experimentation with the ClueWeb09 collection, we observed that the relevance models of top ranked documents appeared to contain many terms that we felt might be misleading because they were particular to the source of the document and did not reflect terms relevant to the topicality of the document. In an attempt to correct for this issue, we first built a relevance model. From this model, we removed all stop words and stop word stems, and then took the 100

Run	P10
WatS.base	0.118
WatS.twen.1	0.222
WatS.fub.1	0.233
WatS.SIEL.1	0.257
WatS.Sab.1	0.259
WatS.twen.2	0.263
WatS.UCSC.1	0.280
WatS.WatS.1	0.306
WatS.WatS.2	0.306

Table 2: Relevance Feedback Results. WatS.base is our baseline without feedback.

most probable terms and for each of those terms, we computed their pointwise KL-divergence:

$$P(w|M_R) \log \frac{P(w|M_R)}{P(w|C)} \quad (1)$$

where w is the word stem, $P(w|M_R)$ is the probability of w given the relevance model M_R, and $P(w|C)$ is the probability of w given a maximum likelihood estimated model (MLE) of the collection.

From these 100 stems, we built an expansion query with the 25 stems with the highest pointwise KL-divergence. We gave each stem a weight equal to the maximum of its pointwise KL-divergence or 0.000001. This ad-hoc weighting scheme looked okay but was not based on any existing practice or theory.

We then mixed the original dependence model query with our expansion query. The dependence model had a query weight of 0.3 and the expansion model had a weight a 0.7.

Our phase 2 runs retrieved from the ClueWeb09 English_[1-10] collection with the parameters of Table 1.

3.3 Results and Discussion

Our phase 1 run, WatS.1 obtained a P5 of 0.440. WatS.2 obtained a P5 of 0.592. When combined, the original run had a P10 of 0.516. We refer to the original, combined run as WatS.1+WatS.2.

Out of 30 phase 1 runs submitted by participants, WatS.2 had the highest P5 (rank 1). The rank 2 submission had a P5 of 0.504. WatS.1 had a rank of 8.

Our phase 2 runs performed significantly worse than our phase 1 run, WatS.1+WatS.2. Nevertheless, as Table 2 shows, our feedback runs did perform better than our phase 2 baseline, WatS.base.

The relatively poor performance of our phase 2 feedback runs could be from several causes:

1. We could have made a mistake in either the indexing or retrieval from English_[1-10] vs. CatB, or there could be some sort of bug in Indri only exposed by the English_[1-10] indexes.

2. The assessors could have changed their criteria of relevance when they later judged the phase 2 runs (and other tracks such as the web track).

3. The documents in CatB could have a much higher prior probability of relevance than the documents in English_[1-10] minus CatB.

4. Since the phase 2 runs are judged over the residual collection, if the phase 1 runs consumed a significant fraction of the relevant documents, then phase 2 runs might perform worse than phase 1.

While at this time we do not have proof, we suspect the answer is #3 above, i.e. CatB has a higher prior probability of relevance.

4 Web Track: Ad-hoc

For the web ad-hoc track, we retrieved from the English_[1-10] collection and employed conventional retrieval techniques to establish various baselines on the ClueWeb09 collection. Our runs were: WatSql, WatSdmrm3, and WatSdmrm3we. Table 1 shows the retrieval parameters.

WatSql is a simple query likelihood retrieval given the query. WatSdmrm3 is a dependence model (DM) plus a relevance model (RM3) as described in section 3.1. WatSdmrm3we used the same queries as the run described in Section 3.1 as WatS.1+WatS.2, which is a DMRM3 run with the RM3 component coming from the top results of the Yahoo and Bing search engines.

4.1 Results and Discussion

Run WatSdmrm3we utilized the same queries as our combined phase 1 RF track run, WatS.1+WatS.2 (see section 3.1). The only difference between the runs is that WatSdmrm3we retrieved against English_[1-10] while WatS.1+WatS.2 retrieved against CatB. The assessors judged WatSdmrm3we to at least a depth of 12 and WatS.1+WatS.2 was judged to a depth of 10. WatS.1+WatS.2 were judged as separate runs as part of phase 1 of the RF track and then WatSdmrm3we was judged at a later date as part of the web ad-hoc track.

The P10 of WatS.1+WatS.2 is 0.516. The P10 of WatSdmrm3we is 0.164. The P10 of WatSdmrm3 is 0.118. The P10 of WatSql is 0.084. Assuming no issues with Indri and no change in assessor judging from phase 1 of the relevance feedback track to the judging of the web ad-hoc track, the DMRM3 web expanded queries of both WatS.1+WatS.2 and WatS-dmrm3we perform significantly better over CatB than over English_[1-10]. Initial inspection of results shows that English_[1-10] minus CatB may contain more spam than CatB.

5 Web Track: Diversity

We submitted three runs to the web diversity track: WatSklq, WatSklfb, and WatSklfu. For each run we first obtained the 25 highest scoring point-wise KL-divergence stems from the 100 most frequent stems in the top 25 documents retrieved by a query likelihood retrieval ranking of the documents containing all of the query terms. We exclude the query stems and 418 stopwords from being part of this set. We used this set of stems in three different ways to generate our three runs. All runs were over the English_[1-10] subset. We next describe in more detail the methods used for our runs.

5.1 Methods

We generated 25 stems following the method explained in Section 3.2 except that we used the top 25 documents returned by a query likelihood retrieval using the query terms and requiring that all returned documents contain all query terms.

For WatSklq, we used the stems to generate 25 additional retrievals. To generate the 25 retrievals, for each of the 25 stems, we appended the stem to the topic's query to generate a new query. The query was thus of the form #filreq(#band(original_query new_stem) #combine(original_query new_stem)) in the Indri query language.

To produce WatSklq, we kept the initial retrieval and these 25 lists in a queue and selected documents from them as follows. The lists are numbered from 0 to 25. List 0 is initially the retrieval of the query alone without any added stems. We repeat the following steps until we have enough results or have emptied all lists. First, we select as our current document the document at rank 1 of list 0. The current document is added to our run's results. We then move list 0 to the end of the queue. For lists at queue positions 0 to 24, we stable sort them in descending order by the rank at which we find the currently selected document. If a list does not contain the document, the document is assumed to be at rank infinity. We remove from each list the current document. When a list has no remaining documents, it is removed from the queue.

The motivation behind this method is to retrieve documents containing different relevant nuggets. The hope is that each list will have a different set of relevant documents. We prefer documents at the top of the lists and we prefer as the next list the list that is most different from the current document. Our "similarity measure" between the current document and a list is the document's rank in the list.

For WatSklfb, we used the 25 stems to expand the query. The expansion terms' relative weights were equal to their pointwise KL-divergence or 0.000001 if the pointwise KL-divergence was negative. The original query had a weight of 0.3 and the expansion terms' query was given a weight of 0.7. Our original query was of the form #filreq(#band(query) #combine(query)) in the Indri query language.

For WatSklfu, we generated 25 additional retrievals in the same manner as for WatSklq. We then used a variant of reciprocal rank fusion [3] to join the original retrieval list and these 25 lists. We gave list 0 (the initial query) a weight of 0.5 and the 25 "query plus stem" runs a weight of 0.5 times the stem's pointwise kl-divergence where negative pointwise kl-divergences were given a value of 0.000001. Then documents were scored based on the reciprocal rank fusion with each list contributing to the score of the document the value:

$$\frac{w}{r + 1 + k} \tag{2}$$

where w is the weight of the list, r is the rank of the document, and $k = 60$. The 1 in the divisor sum appears to be a coding mistake that we overlooked.

5.2 Results and Discussion

The official measure for the diversity track is α-nDCG@10, which is a measure that evaluates a result list on its ability to satisfy multiple information needs with novel and diverse results [2]. All of our runs failed to perform well. WatSklfu, WatSklfb, and WatSklq have average α-nDCG@10's of 0.057, 0.077, and 0.090 respectively. In general, our runs either produced what might be considered an acceptable score for a topic or produced a score of zero or near zero. Our best run, WatSklq, scored at or above the median on 29 out of 50 topics. While WatSklfb did not perform well overall, it did obtain the high score for topic 44, "map of the united states."

Our results may have been hurt by our poorer performance on the English_[1-10] subset compared to CatB (see Section 3.3).

6 Conclusion

While indexing the ClueWeb09 collection in a reasonable amount of time requires a significant amount of hardware, it is feasible with existing retrieval systems such as Indri. By running the same queries on both the CatB subset and the larger English_[1-10] subset, we believe we have discovered that CatB has a higher prior probability of relevance than the remainder of the collection. We plan in future work to examine in more detail the causes of our runs' varied performances.

7 Acknowledgments

Special thanks to David Fisher for his help with indexing ClueWeb09. Iván Patarroyo and Ben Lin wrote and ran the code to query the search engines and download the corresponding web pages.

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC), in part by an Amazon Web Services in Education Research Grant, and in part by the University of Waterloo. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsors.

References

[1] N. Abdul-Jaleel, J. Allan, W. B. Croft, F. Diaz, L. Larkey, X. Li, M. D. Smucker, and C. Wade. UMass at TREC 2004: Novelty and HARD. In E. M. Voorhees and L. P. Buckland, editors, *The Twelth Text REtrieval Conference (TREC 2003)*. Department of Commerce, National Institute of Standards and Technology, 2004.

[2] C. L. Clarke, M. Kolla, G. V. Cormack, O. Vechtomova, A. Ashkan, S. Büttcher, and I. MacKinnon. Novelty and diversity in information retrieval evaluation. In *SIGIR'08*, pages 659–666. ACM, 2008.

[3] G. V. Cormack, C. L. A. Clarke, and S. Buettcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *SIGIR'09*, pages 758–759. ACM, 2009.

[4] R. Krovetz. Viewing morphology as an inference process. In *SIGIR'93*, pages 191–202. ACM, 1993.

[5] V. Lavrenko and W. B. Croft. Relevance based language models. In *SIGIR'01*, pages 120–127. ACM, 2001.

[6] Lemur. Lemur Toolkit for Language Modeling and IR, 2003. `http://www.lemurproject.org/`.

[7] D. Metzler and W. B. Croft. A markov random field model for term dependencies. In *SIGIR'05*, pages 472–479. ACM, 2005.

[8] D. Metzler, F. Diaz, T. Strohman, and W. B. Croft. UMass robust 2005 notebook: Using mixtures of relevance models for query expansion. In *TREC 2005 Notebook*, 2005.

[9] M. D. Smucker. *Evaluation of Find-Similar with Simulation and Network Analysis*. PhD thesis, University of Massachusetts Amherst, 2008.