Ad Hoc and Diversity Retrieval at the University of Delaware

Praveen Chandar, Aparna Kailasam, Divya Muppaneni, Lekha Thota, Ben Carterette
Department of Computer & Information Sciences
University of Delaware, Newark, DE, USA

This is the report on the University of Delaware Information Retrieval Lab's participation in the TREC 2009 Web and Million Query tracks. Our report on the Relevance Feedback track is in a separate document [3].

1 Preprocessing and Indexing

We indexed ClueWeb using the Indri retrieval engine [6]. Due to disk space constraints, we elected to use the Category B subset of 50 million English-language web pages only. We indexed the full documents. We included field information such as title, headings, and bold/italic markup, and dropped script and style tags. We did not index anchor text.

We used the Krovetz stemmer and a simple stopword list. The stopword list is inadequate for the collection, as it does not include meaningless tokens found very frequently in web pages (e.g. http). Attempts to create a more appropriate list by the submission deadline failed.

1.1 Distributed Processing

We built eight separate indexes, each comprising the documents in two subdirectories of the Category B collection. Each index took roughly eight hours to build.

The eight indexes were distributed over eight dual-processor quad-core compute nodes; on each compute node, an Indri daemon process interfaced with the index. Running a query on the head node involved communicating with each of the eight compute nodes: document term statistics could be computed on nodes themselves, and collection statistics were computed by communication between the nodes. After each node scored the documents in its index, they were returned to the head node for merging and output.

With this approach, we could run queries in parallel. Though network communication is a bottleneck, caching collection statistics at each node improved performance. There is more computation involved in calculating the document term statistics (compute-bound) than the collection term statistics (mostly IO-bound), so running eight queries in parallel put us at roughly 40% load over all 64 cores.

By setting up our index this way, we were able to process a query in about half a second on average. Even very long, complicated queries (the result of query expansion or making use of the Indri query language, as shown in Figure 1) ran in less than a second. Thus we were able to run all 40,000 queries for the Million Query Track in a matter of hours.

1.2 PageRank

We calculated PageRank [8] using the web graph information provided with the ClueWeb collection. PageRank calculation was parallelized over 64 cores using Hadoop. We used our own implementation written in python; each iteration took roughly one hour. We completed 10 iterations; while the PageRanks had not yet converged, they did not seem to be changing significantly. We stored the PageRank values in the Indri indexes as document priors, converting each to a log probability.

method	queries run	task	run name
query-likelihood (QL)	1–50; 20051–60000	Million Query ad hoc	udelIndri
QL+dependence model (DM)	1–50; 20051–60000	Million Query ad hoc	udelIndDM
QL+relevance model (RM)	1–50; 20051–60000	Million Query ad hoc	udelIndRM
QL+PageRank prior (PR)	1–50; 20051–60000	Million Query ad hoc	udelIndPR
QL+domain trust prior (SP)	1–50; 20051–60000	Million Query ad hoc	udelIndSP
QL+DM+RM	1–50	Web ad hoc	udelIndDMRM
QL+DM+RM+PR	1–50	Web ad hoc	udelIndDRPR
QL+DM+RM+SP	1–50	Web ad hoc	udelIndDRSP
QL+similarity pruning	1–50	Web diversity	udelSimPrune
QL+facet model (FM) with RM	1–50	Web diversity	udelFMRM
QL+FM with web graph (WG)	1–50	Web diversity	udelFMWG

Table 1: Our Web and Million Query runs: methods used, queries completed, task performed, and run name. Note that Web queries 1–50 were numbered 20001–20050 for the Million Query track, but they are the same topics.

1.3 Domain Trust

We also calculated a simple measure of "trust" in a domain using some publicly-available sendmail and content filtering whitelists and blacklists. Each time a domain appeared on any of our blacklists, we increment its blacklist count b by one. Each time it appeared on any of our whitelists, we incremented its whitelist count w by one. We then assign a trust value $\frac{w+1}{w+b+2}$ to each domain. Finally, we distribute the domain trust values to the individual web pages in that domain and normalize them so that they sum to one. We take the log and store these in the Indri indexes as document priors.

The trust is of course limited by the completeness of the blacklists and whitelists. We originally conceived of this as a "spam prior", but we found that the whitelists provided much more information than the blacklists did.

The 10 most trusted domains were: livejournal.com, homestead.com, suite101.com, live.com, ucla.edu, skyrock.com, stanford.edu, harvard.edu, ca.gov, and army.mil. Some of these are highly trusted solely because of how often they appeared on whitelists, which may actually reflect the opposite of what we intend: livejournal.com may appear on whitelists very often because it actually has more spam than other domains with equivalent numbers of pages. We have not investigated this hypothesis further. Wikipedia was the 289th-most-trusted domain, only because it appeared fewer times on whitelists than other domains—it never appeared on a blacklist.

The 10 least trusted domains were: top-site-list.com, k2free.com, 1sweethost.com, net.cn, toptenreviews.com, 8n.com, oymap.com, co.cc, com.cn, and skydrive.live.com.

2 Retrieval Methods

2.1 Web and Million Query Ad Hoc Tasks

For the ad hoc tasks in the Web and Million Query tracks, we used combinations of the query-likelihood language model (QL), the Markov random field model of term dependencies (DM) [7], and pseudo-relevance feedback with relevance models (RM) [5]. We also used two document priors: the PageRank (PR) and domain trust (SP) priors described above. Table 1 shows all of our ad hoc runs based on combining these methods.

2.1.1 Tuning Parameters

Query likelihood has a Dirichlet smoothing parameter μ. The dependence model has three parameters reflecting the relative weight of the original query, query terms appearing in ordered phrases, and query

```
#weight(0.75 #combine( #prior(pagerank)
                       #weight( 0.8 #combine( obama family tree )
                                0.1 #combine( #1( family tree )
                                              #1( obama family )
                                              #1( obama family tree ) )
                                0.1 #combine( #uw8( family tree )
                                              #uw8( obama tree )
                                              #uw8( obama family )
                                              #uw12( obama family tree ) )))
        0.25 #weight( 0.147 family 0.124 on 0.112 obama 0.106 tree 0.106 2008
                      0.102 trivia 0.079 search 0.078 1 0.077 news 0.069 comment) )
```

Figure 1: Automatically-generated QL+DM+RM+PR query for topic #1 "obama family tree".

terms appearing in unordered windows. The relevance model has four: number of top-ranked documents to use, number of terms to include in the model, smoothing with collection, and smoothing with original query. To set these, we did a simple parameter sweep, running the 784 queries from the 2008 Million Query track on a GOV2 index [1]. We evaluated using the Million Query evaluation measure expected MAP (eMAP) and chose the parameters with greatest eMAP.

We did not have time for a full parameter sweep, so we tried a few values that have been shown to work well in published work along with some other values further away. This gave us a very rough map of the parameter space that we can use to do more detailed tuning in the future.

Figure 1 shows a typical example of our longest query type: the QL+DM+RM+PR model post-expansion. All other types are essentially formed from subtrees of this type.

2.2 Web Diversity Task

We compared two basic models for diversity retrieval: a simple similarity-based pruning method and a more complicated method that probabilistically models "facets" or subtopics (FM) [2]. The subtopic models in FM can be estimated in different ways; we tried an approach based on document relevance models (RM) and one based on link graphs induced from ranked results (WG). Table 1 shows these three runs.

2.2.1 Greedy Result Set Pruning

Greedy result set pruning is a method for pruning documents from a baseline ranking that are likely to overlap in content with other documents in the same ranking. First, the documents are ranked in order of their query-likelihood score. Second, documents that are most similar to previously ranked documents are pruned. Here, we simply step down the ranked list of documents and prune documents with similarity greater than some threshold θ, i.e at rank i, we remove any document D_j, $j > i$, with $sim(D_j, D_i) > \theta$. The similarity score is cosine similarity with TF-IDF term weights.

2.2.2 Probabilistic Set-Based Approach

In the set-based approach we retrieve a set of documents that maximizes the likelihood of capturing all the facets. Consider a set of subtopics F of size m. Then the probability that all of the subtopics are captured by a set of documents D of size n is:

$$\begin{aligned}P(F \in D) = \prod_{j=1}^{m} P(F_j \in D) &= \prod_{j=1}^{m} P(F_j \in D_1 \vee F_j \in D_2 \vee \cdots \vee F_j \in D_n) \\ &= \prod_{j=1}^{m} 1 - \prod_{i}^{n} (1 - P(F_j \in D)).\end{aligned}$$

Note we have assumed that facets occur in documents independently. A diversity retrieval system using this model must do three things. First, it must hypothesize a set of subtopics F. Second, for each subtopic F_j, it must estimate the probability $P(F_j \in D_i)$ that it occurs in each document D_i. Third, it must have a way to select the smallest set of documents that is most likely to contain all the subtopics.

We have two separate ways to hypothesize subtopics and estimate containment probabilities; they are described below. Given all the containment probabilities, we select the set of documents to rank by taking $\max_i P(F_j \in D_i)$ for each subtopic F_j; we then rank these in decreasing order of their original query-likelihood scores. This can be seen as an approximation to maximizing the likelihood of $P(F \in D)$ [2].

Relevance Models. A relevance model is a distribution of words $P(w|R)$ estimated from a set of relevant or retrieved documents. We will model diversity by estimating m subtopic models $P(w|F_1)..P(w|F_m)$ from a set of retrieved document using the RM2 approach described by Lavrenko and Croft [5]. If we knew which documents were relevant to each subtopic, we would estimate each model as:

$$P(w|F_j) \propto P(w) \prod_{f_k \in F_j} \sum_{D_i \in D_{F_j}} P(f_K|D_i)P(w|D_i)P(D_i)/P(w)$$

where D_{F_j} is the set of documents relevant to subtopic F_j, f_k are the terms in the subtopic documents, $P(w) = \sum_{D_i \in D_{F_j}} P(w|D_i)P(D_i)$, and $P(w|D_i)$ is a smoothed estimate.

Since we do not know the relevant documents, we estimate the model from the retrieved documents. Specifically, m models are obtained from the top m retrieved documents by taking each document along with its k nearest neighbors.

Given these relevance models, we estimate containment probabilities $P(F_j \in D_i)$ by transforming $P(D_i|F_j)$ to a binary probability between 0.25 and 0.75 by a simple linear mapping.

Web Graph Model. Network structure can provide information about the topical content of documents: documents that have dense linkings among each other are more likely to be relevant to a subtopic than other documents outside that subgraph. In this method we are interested in finding several densely linked subsets of documents in the results of an initial query-likelihood retrieval. Each subset could potentially cover a different subtopic for a given query.

The top k initial results are represented as a directed graph $G = (V, E)$: nodes correspond to pages and a directed edge $(p, q) \in E$ correspond to the presence of link from p to q. We expand the subgraph to include all the *in-links* and *out-links* to and from the subgraph. The *hubs* and *authorities* scores are calculated using the iterative procedure described by Kleinberg [4].

We then build m models from the set of documents from the top $m - 1$ non-principle eigenvectors and the principle eigenvector. Subtopic models are built from the set of documents as done in the relevance modeling method.

3 Experiments and Results

3.1 Ad Hoc Results

Table 2 shows ad hoc results (precision at 10 from `trec_eval` and two MAP estimates) for all of our Web and MQ track runs (including diversity runs). The three measures have a fair amount of agreement: dependence modeling and the domain trust prior are useful; relevance modeling is useful in conjunction with dependence modeling but much less so on its own; the PageRank prior actively hurts results; diversity retrieval methods do not do well at ad hoc retrieval. There is one obvious outlier: the DM+RM+PR run is far overrated by eMAP compared to P10 and statMAP. Based on the individual DM, RM, and PR results, we are inclined to believe that P10 and statMAP underrated this run's performance while eMAP overrated it, and it should probably by somewhere in the middle.

Some caveats about this table: Million Query results are averaged over all judged queries, which may not be entirely appropriate; the next section shows MQ results broken out into two separate groups. Both

run	task	queries 1–50			20001–60000		
		P10	statMAP	eMAP	P10	statMAP	eMAP
udelFMWG	diversity	0.1480	0.0711	0.0491	–	–	–
udelIndPR	ad hoc	0.2460	0.1194	0.0534	0.1157	0.1588	0.0560
udelSimPrune	diversity	0.3480	0.0753	0.0607	–	–	–
udelFMRM	diversity	0.2600	0.1572	0.0638	–	–	–
udelIndri	ad hoc	0.2980	0.1711	0.0660	0.1525	0.2046	0.0693
udelIndRM	ad hoc	0.2900	0.1745	0.0662	0.1194	0.1619	0.0688
udelIndSP	ad hoc	0.3200	0.1682	0.0767	0.1568	0.2050	0.0746
udelIndDM	ad hoc	0.3320	0.1836	0.0774	0.1606	0.2214	0.0750
udelIndDMRM	ad hoc	0.3260	0.2111	0.0865	–	–	–
udelIndDRPR	ad hoc	0.2020	0.1284	0.0909	–	–	–
udelIndDRSP	ad hoc	0.3380	0.2202	0.0977	–	–	–

Table 2: Ad hoc evaluation results for all ad hoc and diversity runs, sorted by eMAP on queries 1–50. As expected, diversity runs score poorly by ad hoc measures. The DM+RM+SP model seems to be best for queries 1–50, and the DM model is best over all judged queries.

run	baseline			reusability		
	P10	statMAP	eMAP	P10	statMAP	eMAP
udelIndPR	0.1232	0.1693	0.0577	0.0882	0.1543	0.0546
udelIndRM	0.1267	0.1732	0.0705	0.0903	0.1467	0.0628
udelIndri	0.1553	0.2123	0.0720	0.1382	0.2081	0.0652
udelIndSP	0.1593	0.2110	0.0776	0.1414	0.2110	0.0708
udelIndDM	0.1657	0.2362	0.0799	0.1473	0.2189	0.0696

Table 3: Million Query results broken out by baseline queries (that we contributed judgments to) and reusability queries (that we did not). There were 405 baseline queries and 186 reusability queries.

statMAP and P10 estimates change substantially from the first 50 queries (which averaged 262 category B judgments each) to the remaining 39,950 (which averaged 50 category B judgments each); since we did not do anything different for those queries, we must conclude that this is a function of having fewer judgments available to compute estimates. The eMAP estimates are virtually the same in the two groups. P10 results should be taken with a grain of salt, as not all documents in the top 10 were judged (this is especially true of the Million Query queries). In particular, it is strange that the diversity run udelSimPrune did the best of all runs by precision at 10, but was among the worst by MAP estimates.

Nevertheless, it seems clear that the DM+RM+SP model is best among these, and that DM and SP are both generally useful for retrieval in the general web. All measures agree that the diversity runs performed poorly, which is not unexpected considering they were not designed to perform ad hoc retrieval.

3.1.1 Million Query Results

The Million Query track included a reusability study in which sites were held out of a subset of queries. Evaluation results over those queries may differ from evaluation results over the queries the site contributed to. Table 3 shows results for our five MQ runs broken out by "baseline" queries that we contributed to (405 queries total) and "reusability" queries that we did not (186 queries total). The ranking of systems changes slightly between the two, but not in a way that would refute our conclusions above.

3.2 Diversity Results

Figure 2 shows a plot comparing our diversity results to the median for each query. In general they did not perform particularly well. The simple similarity-pruning approach was our best run by αNDCG@10.

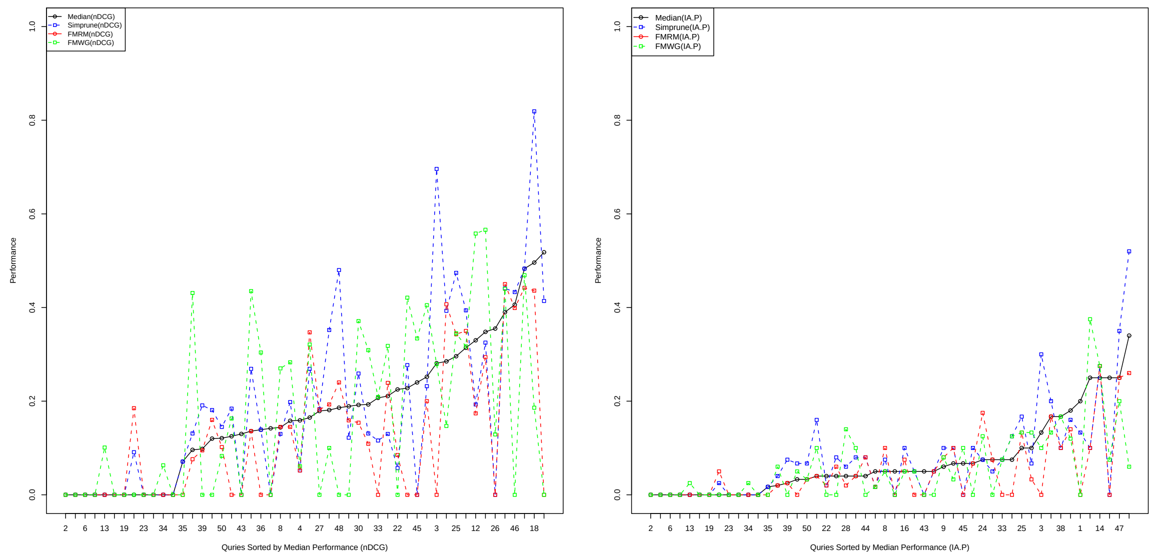

Figure 2: Diversity run results compared to the median performance (black line) for each query. Our three methods seem to have distinct strengths and weaknesses, as evidenced by that fact that they do not coincide very much.

Diversity results for all 11 runs are shown in Table 4. By αNDCG@10, two of our diversity runs outperformed any of our ad hoc runs. By IA-P10, the similarity-pruning method was in the top 3—the fact that our IA-P10 performance was worse than our αNDCG@10 performance suggests that a small number of intents may have dominated, and that the best ad hoc approaches were picking up on those at least as well as the diversity approaches. Despite not performing well compared to the median for the task, generally performed well compared to ad hoc runs, suggesting that even "broken" approaches to diversity can improve on the diversity provided by traditional retrieval methods.[1]

It is interesting that our diversity methods did so well compared to the ad hoc methods even though they found many fewer relevant documents (as the ad hoc results show). This suggests to us that there is a great deal of redundancy in this corpus, and that novelty and diversity are therefore very worthy subjects of future research.

However, we note that few of the differences are statistically significant. Among the diversity runs, only the 50% improvement from udelFMRM (αNDCG10 0.126) to udelSimPrune (αNDCG10 0.189) is significant by a paired t-test at the $\alpha = 0.05$ level. The 33% improvement from udelFMRM to udelFMWG is not significant. This is due to very high variance in per-query performance. When all 11 runs are considered together (and the p-values adjusted according to Tukey's Honest Significant Differences (HSD) procedure to ensure a family-wise false positive rate of 0.05), there is not enough evidence to truly find any of the pairwise differences significant. The same is true of the IA-P10 results.

3.2.1 Diversity for Relevance Feedback

Though our main Relevance Feedback track experiments are described in a separate paper [3], we will briefly describe one diversity-related experiment we did for that track. One of our Phase 1 submissions—the `udel.2` run—used our similarity-based pruning method to try to find five documents that were likely to be on different topics. The hypothesis is that documents that distinguish between query intents may provide

[1]Our implementation of the FMRM model had a bug resulting from a misunderstanding of how Indri's distributed processing worked; we believe this bug explains its poor performance.

run	task	αNDCG10	IA-P10
udelIndRM	ad hoc	0.102	0.056
udelIndDRPR	ad hoc	0.116	0.049
udelIndri	ad hoc	0.124	0.061
udelFMRM	diversity	0.126	0.052
udelIndDM	ad hoc	0.140	0.073
udelIndSP	ad hoc	0.141	0.071
udelIndDRSP	ad hoc	0.141	0.086
udelIndPR	ad hoc	0.146	0.055
udelIndDMRM	ad hoc	0.152	0.083
udelFMWG	diversity	0.168	0.060
udelSimPrune	diversity	0.189	0.081

Table 4: Diversity evaluation results for all 11 runs sorted by αNDCG10.

better feedback.

We believe we can count that hypothesis as falsified (to the extent that similarity pruning does what it's supposed to): the `udel.2` judgments provided better results than other judgments only 19% of the time, compared to our `udel.1` judgments that outperformed others 82% of the time.

4 Conclusions

Based on our results, we draw the following conclusions:

- standard retrieval models can work reasonably well on the general web;
- even computationally-intensive models like relevance models can be computed reasonably quickly with distributed processing;
- PageRank does not seem to be a useful feature, but some measure of domain trust does;
- ad hoc retrieval methods do not generally work well for diversity retrieval, and vice versa;
- decent diversity results do not seem to rely on finding a lot of relevant material;
- a simple similarity-pruning approach seems to give decent diversity results.

Additionally, the results seem to point to a conclusion about Million Query evaluation methods: eMAP is more robust to differing numbers of relevance judgments than statMAP.

References

[1] James Allan, Javed A. Aslam, Ben Carterette, Virgil Pavlu, and Evangelos Kanoulas. Million Query Track 2008 overview. In *Proceedings of TREC*, 2008.

[2] Ben Carterette and Praveen Chandar. Probabilistic models of novel document ranking for faceted topic retrieval. In *Proceedings of CIKM*, 2009.

[3] Ben Carterette, Praveen Chandar, Aparna Kailasam, Divya Muppaneni, and Lekha Thota. Minimal test collections for relevance feedback. In *Proceedings of TREC*, 2009.

[4] Jon Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[5] Victor Lavrenko and W. Bruce Croft. Relevance based language models. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 120–127, 2001.

[6] Donald Metzler and W. Bruce Croft. Combining the language model and inference network approaches to retrieval. *Information Processing and Management*, 40(5):735–750, 2004.

[7] Donald Metzler and W. Bruce Croft. A markov random field for term dependencies. In *Proceedings of the 28th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR)*, pages 472–479, 2005.

[8] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web, 1999.