University of Twente @ TREC 2009: Indexing half a billion web pages

Claudia Hauff and Djoerd Hiemstra

University of Twente, The Netherlands
{c.hauff, hiemstra}@cs.utwente.nl

– draft –

1 Introduction

The University of Twente participated in three tasks of TREC 2009: the adhoc task, the diversity task and the relevance feedback task. All experiments are performed on the English part of ClueWeb09.

We describe our approach to tuning our retrieval system in absence of training data in Section 3. We describe the use of categories and a query log for diversifying search results in Section 4. Section 5 describes preliminary results for the relevance feedback task.

2 ClueWeb09

The English part of ClueWeb09 (ClueWeb09_English_1 to ClueWeb09_English_10) was indexed with the Lemur Toolkit, version 4.9. We did not attempt to index the non-English part. The documents were Krovetz stemmed. The indexed metadata fields are *title*, *url* and *heading* (html tags H1 to H4).

2.1 Anchor texts

We extracted the anchor text and indexed it separately. Due to time constraints, we were only able to match the anchor text within each of the 10 sub corpora, that is anchor text found in a document of English_2 linking to a document in English_5 was not considered. This resulted in anchor texts for 297 million docmentes, which is almost 59 % of the English part of ClueWeb09. Full anchor text extraction, which we ran after the official submission, resulted in anchor texts for about 87 % of the English web pages.

2.2 Spam Detection

A collection crawled from the Web is likely to contain a considerable amount of spam pages. Many approaches exist to identify spam pages based on page content, hyperlink structure, URL form, the similarity between pages of a single host and combinations of those features [1, 2, 3, 4]. For the TREC experiments we implemented a basic spam detection mechanism, that relies on page content, page title features and URL form.

In a first step, we determined the number of different hostnames in the corpus. In total, we found more than 18 million hostnames among the English part of ClueWeb09, many with 1 or 2 pages only. While 47.78% of hosts contain 1 page only, together those pages amount to only 1.71% of all pages in the corpus. A total of 99.58% of hosts contain less than 1000 pages, and 96.75% of hosts have less than 100 pages. We concentrated our efforts of detecting spam on the small percentage of hosts with 100 or more pages in the English part of the ClueWeb09 corpus. The total number of those hosts is 589,343. Together, they make up 75.02% of all pages in the corpus. Each of the hosts with 100 or more pages is classified as either a spam host or a no-spam host. This result is then used to postprocess the retrieval results – all documents belonging to a host classified as spam are filtered out and lower ranked results are used to fill the gap. When a document belongs to a host that was not classified it is also filtered out.

Problematic is the lack of training data for ClueWeb09. We relied on the judgments of the Web Spam Challenge in 2006 and 2007[1], where a corpus of documents (UK domain) was labelled as spam/no-spam on the host level. When combining the training and test data provided, we find 296 spam hosts in ClueWeb09 with ten or more pages. If the minimum page count per host is set to 100 pages, we find 136 spam hosts. To boost the number of positive (i.e. spam) training examples, we manually jugded sampled hostnames from the ClueWeb corpus and found another 75 spam hosts, thus in total we have 371 positive spam examples with 10 or more documents. During the judging process we also identified 442 negative (no-spam) training examples.

The document features we rely on, include the document length, the title length, the ratio between plain text and html text, the ratio between plain text and anchor text, the number of digits in a URL, etc. For hosts with more than 100 pages in the corpus, we sample 100 pages and determine a total of 28 features for each sampled document. Then, as host features we rely on the mean and the standard deviation of each feature, resulting in a total of 56 features for each host. Given the features and the training examples we use Weka and its C4.5 implementation (called J48) to derive a decision tree. Two variations are tested, namely J48 and a cost-sensitive variety of J48 that punishes the error spam host classified as no spam 5 times as much as vice versa. The latter classifier is thus more strict, we will denote it CS5.

To evaluate both algorithms, we performed 10-fold cross-validation. For the J48 approach, 78.23% of the instances are classified correctly. The CS5 approach

[1] `http://webspam.lip6.fr`

	classified as	
	NO-SPAM	SPAM
NO-SPAM examples	353	89
SPAM examples	88	283

(a) J48

	classified as	
	NO-SPAM	SPAM
NO-SPAM examples	305	137
SPAM examples	54	317

(b) CS5

Table 1: Confusion matrices of spam versus no-spam examples based on 10-fold cross-validation.

yielded 76.51% correctly classified. Though CS5 leads to less correctly classified examples, it filters out more spam hosts as can be seen in the confusion matrices of Table 1. We tested both approaches in our submitted runs. Given the trained decision tree, all hosts with 100 or more pages in the corpus were classified as either spam or no-spam. For the J48 approach, out of 589,343 hosts, 440,043 (74.66%) hosts were found to be no-spam, 25.34% were identified as spam. Conversely, for CS5, 351,307 (59.61%) were labeled as no-spam, while 40.39% were identified as spam hosts.

3 Adhoc Track

In the adhoc task, we rely on an automatic evaluation approach proposed by Soboroff et al. [6], called Random Sampling (RS) to select the best retrieval strategy from a mix of retrieval approaches. We derive a number of retrieval runs, based on different retrieval methods such as Okapi, Language modeling, but also based on different indexed fields, e.g. the title field, the anchor text index, etc. Then, RS is employed to estimate which of those systems performs best. In previous work, RS was found to perform well to find the poor and average systems, while severely misjudging the best systems. We found in experiments with more diverse and more recent data sets, that this is only a significant problem in cases where manual systems are included in the pool of systems.

Apart from estimating the best retrieval method, we also face the issue of merging the results. The indices are on 10 different machines, thus at one point we need to merge the ranked lists of results. In Figure 1 the overall system is depicted. Each query is sent to the 10 indices C_1 to C_{10} and m different retrieval methods are applied, resulting in m different result lists. Then, spam detection is applied to each ranked list and those documents, determined to be spam are filtered out and removed from the list. We deal with each set of result lists separately, RS is applied to estimate which of those is the best performing one. Then, the 10 result lists are merged to form the final result list.

Merging was tested in three flavours: (i) normalizing of retrieval scores by ZMUV (zero mean, unit variance) and merging of the results based on the

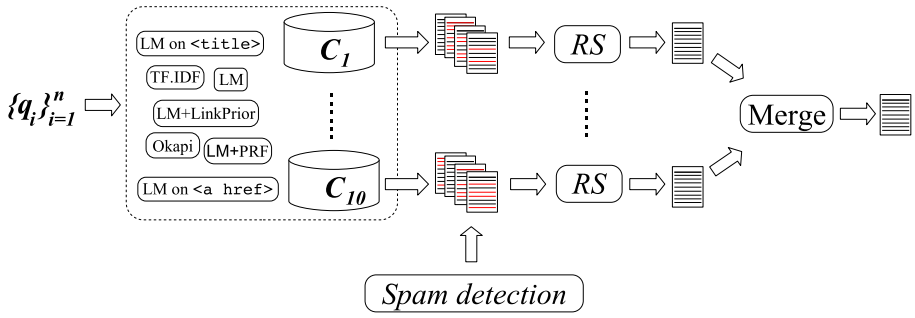

Figure 1: Retrieval chain of the adhoc task.

normalized scores, (ii) round robin merging, by first taking the top result of each index, then the second of each index and so on to possibly diversify the results and (iii) to merge the unnormalized scores.

In order for *RS* to be useful, there should be a relatively large number of retrieval runs to rank, but also the runs should differ in their results. For instance, it may not be very useful to derive 20 Okapi runs, that only differ in their parameter settings. We attempted to derive a variety of retrieval runs. As basic retrieval approach we rely on Dirichlet smoothed language modeling. We derived 28 runs for each of the 10 corpora including runs based on document title, anchor text, non-uniform document priors, pseudo-relevance feedback, etc.

Submitted as official runs were the runs listed in Table 2. For *twCSrs9N* the results are filtered according to CS5 and ZMUV merging is employed. For run *twCSrsR* merging was performed in a round robin manner. Merging without normalization and J48 spam detection was tested in run *twJ48rsU*. Round robin merging results in the best mean average precision, but merging without normalization results in better early precision.

runID	map	bpref	P@5	P@100	eMAP	eP@5	eP@100
twCSrs9N	0.0297	0.1480	0.2080	0.0758	0.0217	0.2250	0.2421
twCSrsR	0.0403	0.1700	0.2000	0.0960	0.0263	0.2200	0.2570
twJ48rsU	0.0284	0.1211	0.2760	0.0742	0.0200	0.2800	0.2501

Table 2: Overview of submitted adhoc runs.

4 Web Diversity Track

In the diversity task, we rely on categories and a query log to determine topic aspects.

4.1 Category & query log information

We derived a list of 98 categories from the Open Directory Project (ODP),[2] mostly consisting of single terms only, such as *education*, *animal*, etc. Then, we relied on Indri's Boolean AND operator, *#band*, to force the retrieval system to only return documents containing all query terms and the category term. This step was performed for each combination of query and category term(s). A ranking of category terms was determined by sorting them according to retrieval score of the top retrieved document. The five categories with the highest retrieval score were kept. For instance, for the query *solar panel* (wt09-45) and the subcorpus English_1, the top five categories found are *product*, *company*, *environment*, *video* and *research*.

As an alternative source of information, we used a query log consisting of over 21 million queries [5]. To respect the anonymity of the users, we used a stripped version of the query log that only contains the actual queries issued in random order (and none of the other metadata, such as user id and clicks). Each query was located in the log and the terms that co-occur with the query most often were determined. For instance, the query *dinosaurs* (wt09-14) occurred in the log as *dinosaurs* (194 times), *dinosaurs com* (12), *walking with dinosaurs* (9), *www dinosaurs com* (7), *the dinosaurs* (7), *dinosaurs tv show* (6), *dinosaurs pictures* (6), *types of dinosaurs* (6), *dinosaurs for kids* (6) and *dinosaurs the tv show*.

From each of the 50 queries, a number of new queries are derived, containing the original query string and one of the top five category terms and the query log terms respectively. Retrieval was performed for each query; subsequently those retrieval results were merged. Here, we performed round robin merging, as we do indeed want to cover different categories. Score normalization with subsequent merging might not bring about the desired results, if a particular aspect of a query has too low scores. This process is repeated for each corpus English_1 to English_10. In a final step, the results over the 10 corpora are merged.

4.2 Experimental results

The two runs officially submitted are described in Table 3. Spam detection was again used as a preprocessing step, the result merging occurred either in a round robin fashion (RBB) over the 10 subcorpora or according to normalized scores (RNB).

runID	α-ndcg@5	α-ndcg@10	α-ndcg@20	IA-P@5	IA-P@10	IA-P@20
twCSodpRNB	0.123	0.145	0.167	0.062	0.063	0.056
twCSodpRBB	0.144	0.164	0.193	0.067	0.062	0.061

Table 3: Overview of submitted TREC runs to diversity task.

[2] http://www.dmoz.org

5 Relevance Feedback Track

The Relevance Feedback track consists of two phases. In the first phase, for each query, five documents are identified that are judged by an assessor. In the second phase then, the knowledge gained of the (non-) relevance of those documents shall be exploited. For both phases the Category B data set was used.

5.1 Phase 1

We relied on Indri's query language for a baseline run with combinations of keywords, phrases and unordered windows of size 8 and 12. The top five documents, from different domains, retrieved per query were used. Thus, the only filtering performed was according to URL in order to avoid returning five very similar documents from the same domain. This run is *twen.1*. Of the top five documents of the 50 queries, fourteen had not a single relevant document among them while three had not a single non-relevant document among them. Thirteen queries had at least one highly relevant document.

The second submitted run *twen.2* is a keyword only Dirichlet smoothed language modeling run ($\mu = 1500$), without domain filtering applied. Sixteen queries had not a single relevant document in the top five documents, two queries had not a single non-relevant document and twelve queries had at least one highly relevant document among them.

5.2 Phase 2

Given a number $d = \{10, 50, 100, 250, 500\}$ of top retrieved documents from the baseline retrieval approach (LM with Dirchlet smoothing: $\mu = 1500$), the $t = \{5, 10, 20, 50, 100\}$ most discriminative terms in them are determined. For each parameter combination the expanded query is formed by adding the t most discriminative terms to the original query. Retrieval is performed, yielding runs r_1 to $r_{t \times d}$.

Given the relevance judgments of five documents for each query, the quality of each r_i is determined in an adhoc fashion. The relevance of a document is either 2 (highly relevant), 1 (relevant) or 0 (not relevant). For each relevance judgment, the rank of the document in the run's top 1000 retrieved documents is determined. If the docid is not found, a value of 0 is assigned, if it is a highly relevant document, its score is $2 \times (1000 - \mathit{rank})$, if it is relevant $(1000 - \mathit{rank})$ and if it is not relevant $-0.5 \times (1000 - \mathit{rank})$. A run that does not contain any of the five judged documents is considered to be better than a run that contains one or more non-relevant documents but no relevant one.

The run of the $t \times d$ runs that has the highest quality for a given set of relevance judgments is the run used for that particular query.

5.3 Experiments

Spam detection (CS5) was performed as a post-processing step. Reranking by document length was switched on for *twen.ugTr.1*, *twen.twen.1* and *twen.UCSC.2*. The results of the submitted runs are described in Table 4. Each run is created by selecting the best (according to the given relevance judgments) expanded query run out of 25 possible ones.

runID	**emap**	**stAP**
twen.twen.1	0.0317	0.1528
twen.twen.2	0.0306	0.1344
twen.ilps.1	0.0369	0.1481
twen.FDU.1	0.0311	0.1415
twen.YUIR.2	0.0398	0.1384
twen.ugTr.1	0.0327	0.1297
twen.UCSC.2	0.0343	0.1294

Table 4: TREC runs submitted to the relevance feedback task.

6 Discussion

This draft report presents perliminary results for the TREC adhoc task, the diversity task, and the relevance feedback task. We investigated, amongst others, unsupervised tuning of our system and the use of categories and query log information for diversifying search results. Additional experimental results will be described in the final version of this paper, showing if the approaches were indeed successful.

To deal with the new ClueWeb09 collection, we needed to aquire at least two skills that are unnessesary for tests on previous TREC collections. First, the data, even if only the English part is used, cannot be indexed on a single machine, requiring some cluster computing skills. Second, web spam seems to be a much bigger problem in ClueWeb09 than in previous (web) TREC collections, requiring some spam detection skills.

Acknowledgements

The research was funded in part by MultimediaN. We are greatful to Yahoo Research, Barcelona, for contributing to our Hadoop cluster.

References

[1] C. Castillo, C. Corsi, D. Donato, P. Ferragina, and A. Gionis. Query-log mining for detecting spam. In *AIRWeb '08: Proceedings of the 4th inter-*

national workshop on adversarial information retrieval on the web, pages 17–20, 2008.

[2] C. Castillo, D. Donato, A. Gionis, V. Murdock, and F. Silvestri. Know your neighbors: web spam detection using the web topology. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval*, pages 423–430, 2007.

[3] D. Fetterly, M. Manasse, and M. Najork. Spam, damn spam, and statistics: using statistical analysis to locate spam web pages. In *WebDB '04: Proceedings of the 7th International Workshop on the Web and Databases*, pages 1–6, 2004.

[4] A. Ntoulas, M. Najork, M. Manasse, and D. Fetterly. Detecting spam web pages through content analysis. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 83–92, 2006.

[5] G. Pass, A. Chowdhury, and C. Torgeson. A picture of search. In *InfoScale 06: Proc. of the 1st international conference on Scalable information systems*, 2006.

[6] I. Soboroff, C. Nicholas, and P. Cahan. Ranking retrieval systems without relevance judgments. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on research and development in information retrieval*, pages 66–73, 2001.