# Microsoft Research at TREC 2011 Web Track

Bodo Billerbeck, Nick Craswell, Dennis Fetterly, Marc Najork

Microsoft

**Abstract**

This paper describes our entry into the TREC 2011 Web track. We extracted and ranked results from the ClueWeb09 corpus using a parallel processing pipeline that avoids the generation of an inverted file. We describe the components of the parallel architecture and the pipeline, how we ran the TREC experiments, and we present effectiveness results.

## 1 Introduction

This paper describes our entry into the TREC 2011 Web track. Our team comprised members from Bing, Microsoft's search engine, and Microsoft Research, Silicon Valley. We used the DryadLINQ data-parallel processing system [10] on a cluster of about 220 machines to process the half-billion page English-language subset of the ClueWeb09 collection. We also used the Scalable Hyperlink Store [8] running on a cluster of 12 machines to compute SALSA scores for each topic. Our retrieval and ranking pipeline consisted of multiple stages. In the first stage we generated query reformulations, using three different rewriting techniques, all based on the anchor text of co-citing hyperlinks. In the second stage we retrieved results for the original topic, and separately for each reformulation, from the ClueWeb corpus. In the third stage, we computed five features (described below) for each candidate result. In the final stage we used a weighted linear combination to blend these features in each result set followed by a second weighted linear combination to blend the scores of the result sets associated with the original and rewritten topics. We performed four runs, and submitted two of them to both the ad-hoc and the diversity task, and the other two runs to a single task each.

## 2 Processing Pipeline

The processing pipeline we used to prepare this year's entry consisted of the following steps:

- **Parsing:** We used DryadLINQ to parse the HTML web pages, tokenizing them into individual words and hyperlinks. We associated each title and content word occurrence with the document containing it; in addition, we associated each anchor word occurrence with the document referenced by the anchor's hyperlink.
- **Query Reformulation** We developed 3 rewriting techniques, which are numbered 1,2,3 and described in Section 3. The original queries can be seen as a "baseline" rewriting, with technique number 0.
- **Document Frequencies:** We computed document frequencies for the union of all terms of the queries we were considering. The queries included rewrites of this year's 50 topics, the 100 topics from TREC 2009 and TREC 2010.
- **BM25F score:** We computed a BM25F score for each query and each document in the ClueWeb09 collection, retaining the top 5000 results per query.

- **SALSA score:** Furthermore, we used SHS to run SALSA-SETR [9] on the top 5000 results of each query. SALSA-SETR is a variant of Lempel & Moran's SALSA algorithm [7], which in turn is a variant of Kleinberg's HITS algorithm [6], one of the best-known link-based ranking algorithms.

- **Inter-domain in-degree** We also used SHS to compute the inter-domain in-degree of each result page, that is the number of hyperlinks from pages in different domains referring to the result page.

- **Matching anchor count (MAC):** We resolved all the symbolic hostnames in ClueWeb09 page and link URLs into IP addresses. Then, using DryadLINQ, we identified unique $\langle s, t, a \rangle$ triples, where $s$ is the first three octets of the IP address of a document, $t$ is the target URL of a link within that document and $a$ is the anchor text of that link. Then we built the MAC ranking feature for each query-target pair $\langle q, t \rangle$, counting the number of source IPs that link to target $t$ with anchor $a = q$.

- **Extraction:** For each query we identified a pool of documents and collated our ranking features. The resulting "extraction" file could be used for training or for generating submitted runs.

- **Spam scores:** For each result, we extracted the "spam score" provided by the University of Waterloo[1].

- **Training:** We combined evidence from the various features by applying ad-hoc transformations to each feature (identity or log, depending on the feature) and then performing a weighted linear combination of the transformed features. For the original queries, we trained the weights using an extraction of the 2009 and 2010 topics and their official judgments. Then, having scored each original and rewritten query using "optimal" weights, we performed a weighted linear combination of the scores of each rewriting technique, again training on the 2009 and 2010 judgments.

- **Runs:** Using an extraction of the fifty 2011 topics, we ran our trained model to produce our submitted runs.

This pipeline differed from the approach we took last year [3] in the following ways:

- **Query reformulation:** The main innovation of this year's entry compared to our previous entries is the addition of query rewriting techniques. Commercial engines can employ multiple data sources, including query logs, to drive such rewriting; however we did not want to utilize any proprietary data. The techniques are described in Section 3.

- **Omitted and added features:** We re-introduced the inter-domain in-degree feature that we employed in our 2009 entry [2]. Conversely, we removed the "span score" feature introduced in our 2010 entry since it was not effective.

# 3 Query Rewriting Techniques

Query *rewriting* transforms an input query such as 'obama family tree', into a rewrite query such as 'barack obama genealogy'. Our ranking model this year (Figure 1) combined signals from the original query and three rewrites. Using rewrites, rather than traditional query expansion, allowed us to make use of our MAC ranking feature. MAC is a phrase match between query and anchor, so is defined for a rewrite query, but would not be defined under blind feedback with a weighted bag-of-words expansion.

Following [5] we treated anchor text information as though it were search log data. We generated a bipartite click graph [4] from the ClueWeb99 anchors, using the anchor phrases as queries, which are connected the link target URLs as though they were 'clicked'. For clarity, we will continue to refer to the bipartite graph as being between anchors and URLs.

In rewriting techniques 1 and 2, we generated a set of simple phrase-level translation probabilities. Given that an anchor contains term $A$, the probability of translation $A'$ is the probability of finding another anchor in two steps

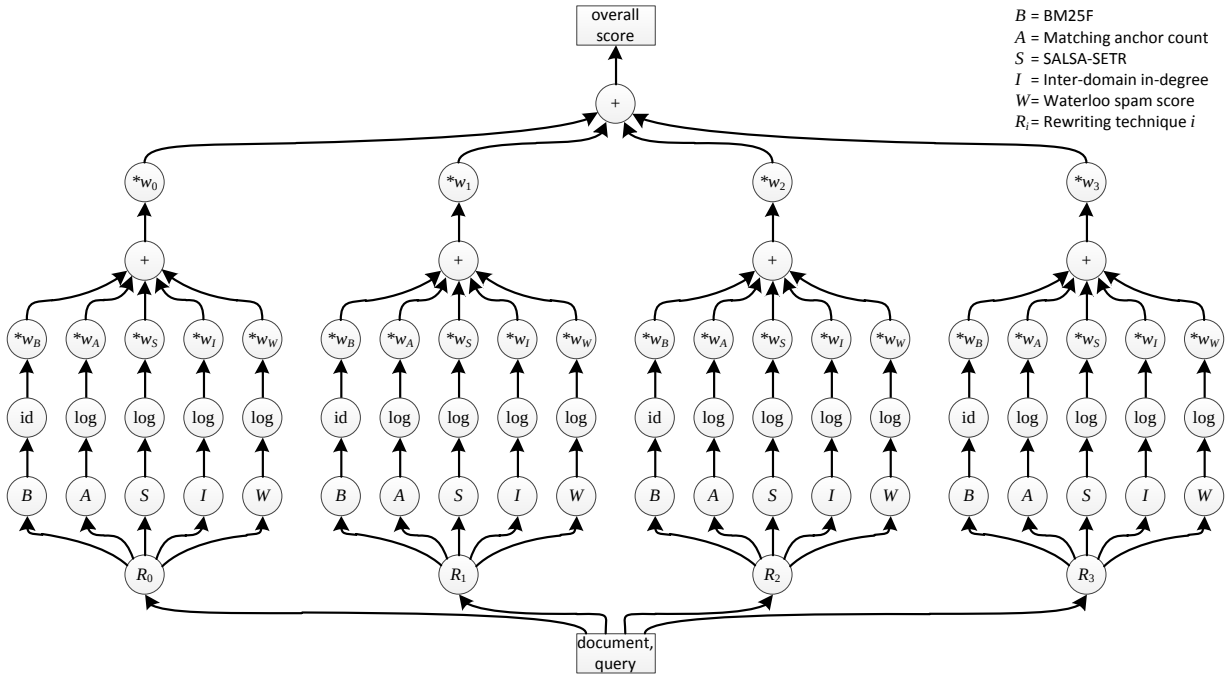


Figure 1: Scoring pipeline

on the bipartite graph where $A$ is replaced with $A'$. For example, the probability of translation 'az' for the term 'arizona' is $0.00575$ because $181,900$ anchors contain 'arizona' and $1046$ of those phrases co-cite a URL with the corresponding 'az' query. To reduce noise, we eliminated translations that happened in fewer than 50 anchor pairs.

In technique 1 we took the most probable translation for each query, so for 'arizona game and fish' would generate the rewrite 'az game and fish'. Another possible translation, from 'game and fish' to 'fish and game' was not applied because it is less probable under the simple translation model.

In technique 2, to further reduce noise we only considered translations where the most probable translation of $A$ is $A'$ and the most probable translation of $A'$ is $A$. For each input query we then applied the translation with the highest $score = max(strlen(A), strlen(A'))$. So translating 'game and fish' to 'fish and game' (13 characters) was chosen, rather than 'arizona' to 'az' (7 characters).

Technique 3 went back to the bipartite graph. To rewrite a query, there must be an anchor in the graph that matches exactly. We then took two steps on the graph to find rewrites, choosing the rewrite that was connected by the most two-step paths, which means having the most URLs in common. To reduce noise we eliminated rewrites with only one path. We also removed rewrites that were simply shortenings of the input query or contained noise terms 'free', 'wikipedia', 'www', 'click', 'here', 'com', 'org', 'site', 'website', 'more', 'link', 'amp', 'lt', 'gt', or 'nbsp'. Technique 3 yielded some of the most interesting rewrites, for example 'to be or not to be that is the question' is rewritten as 'hamlet s soliloquy'.

# 4 Web track submissions

We submitted three runs to the ad-hoc task and three runs to the diversity task. In all six runs the ranker is a simple linear combination of features. Figure 1 shows the ranking tree. The following table shows the weights used in each run. In addition to the six submitted runs, we also report on two "baseline" runs (named *unofficial1* and *unofficial2*) that combine our five features but do not perform query rewriting.

| Run | $w_B$ | $w_A$ | $w_S$ | $w_I$ | $w_W$ | $w_0$ | $w_1$ | $w_2$ | $w_3$ |
|---|---|---|---|---|---|---|---|---|---|
| msrsv2011a1 | 1 | 1.0 | 100 | 0.1 | 1.0 | 1 | 0.05 | 0.000 | 0.20 |
| msrsv2011a2 | 1 | 1.0 | 100 | 0.1 | 1.0 | 1 | 0.00 | 0.060 | 0.22 |
| msrsv2011a3 | 1 | 1.0 | 100 | 0.1 | 1.0 | 1 | 0.10 | 0.150 | 0.05 |
| msrsv2011d1 | 1 | 1.0 | 100 | 0.1 | 1.0 | 1 | 0.00 | 0.060 | 0.22 |
| msrsv2011d2 | 1 | 2.8 | 100 | 0.1 | 3.6 | 1 | 0.00 | 0.016 | 0.00 |
| msrsv2011d3 | 1 | 1.0 | 100 | 0.1 | 1.0 | 1 | 0.10 | 0.150 | 0.05 |
| unofficial1 | 1 | 1.0 | 100 | 0.1 | 1.0 | 1 | – | – | – |
| unofficial2 | 1 | 2.8 | 100 | 0.1 | 3.6 | 1 | – | – | – |

We evaluated training runs using multiple performance measures: for the adhoc task, ERR@20, nDCG@20, P@20, and MAP; for the diversity task, ERR-IA@20, $\alpha$-nDCG@20, NRBP, and MAP-IA. Our training process left us with a pool of runs each of which excelled on a particular measure. We selected our submitted runs such that they did well on the primary measure of each task while also doing well on at least one alternate measure. Run msrsv2011a2 is identical to run msrsv2011d1 and run msrsv2011a3 is identical to run msrsv2011d3.

The next table shows the performance of our six submitted runs and the two unofficial "baseline" runs compared to the median performance of all TREC 2010 web track entries, according to a variety of different performance measures.

| Run | ERR@20 | nDCG@20 | P@10 | P@20 |
|---|---|---|---|---|
| msrsv2011a1 | 0.140 | 0.270 | 0.372 | 0.338 |
| msrsv2011a2 | 0.141 | 0.270 | 0.372 | 0.335 |
| msrsv2011a3 | 0.143 | 0.273 | 0.360 | 0.327 |
| msrsv2011d2 | 0.144 | 0.258 | 0.362 | 0.319 |
| unofficial1 | 0.142 | 0.267 | 0.364 | 0.343 |
| unofficial2 | 0.141 | 0.252 | 0.354 | 0.314 |
| Median | 0.106 | 0.188 | 0.294 | 0.262 |

| Run | ERR-IA@20 | $\alpha$-nDCG@20 | P-IA@20 |
|---|---|---|---|
| msrsv2011d1 | 0.499 | 0.608 | 0.291 |
| msrsv2011d2 | 0.474 | 0.584 | 0.266 |
| msrsv2011d3 | 0.481 | 0.581 | 0.284 |
| msrsv2011a1 | 0.503 | 0.613 | 0.290 |
| unofficial1 | 0.494 | 0.608 | 0.288 |
| unofficial2 | 0.480 | 0.586 | 0.260 |
| Median | 0.408 | 0.516 | 0.232 |

We can draw two observations from these performance numbers: First, all of our runs had better than median performance; and second, query rewriting did not lead to substantial performance improvements. The improvements are not statistically significant.

# 5 Feature Analysis

This section presents a brief analysis of MAC, which is one of our most important ranking features, to show that it performed differently this year than in previous years.

We first analyze how many documents per query have non-zero MAC (its coverage). As indicated in the table, the coverage dropped this year, with the median coverage dropping from hundreds of documents to 7 documents.

| | TREC-2009 | TREC-2010 | TREC-2011 |
|---|---|---|---|
| Minimum | 2 | 0 | 0 |
| Median | 254 | 511 | 7 |
| Maximum | 1 302 506 | 28 284 | 995 |

The explanation for the coverage drop is a change in query sampling this year. In previous years, the average query length was around two words. This year all queries are 3 or more words, and the average length is 3.4 words. MAC is only defined if the query exactly matches an anchor. For longer queries it is less likely that such anchors exist, and therefore MAC has lower coverage.

The second analysis is to confirm that MAC, with its lower coverage, was also relatively less useful as a ranking feature this year. We compare the adhoc performance of a MAC-only ranker to a BM25F-only ranker. We use 2010 and 2011 queries, omitting 2009 since assessors that year used different judging guidelines.

| | Adhoc 2010 | | Adhoc 2011 | |
|---|---|---|---|---|
| | NDCG@20 | ERR@20 | NDCG@20 | ERR@20 |
| BM25F | 0.095 | 0.070 | 0.225 | 0.111 |
| MAC | 0.182 | 0.150 | 0.190 | 0.139 |
| *Relative performance* | *+91%* | *+114%* | *-15%* | *+26%* |

On the 2010 queries, where MAC had better coverage, it also performed relatively better than BM25F, around twice as well, whereas on the 2011 adhoc task the two features perform roughly equally well.

# 6 Conclusions

In preparing our entry this year, we utilized the same computational infrastructure we employed in the previous two years: the DryadLINQ data-parallel processing platform and the Scalable Hyperlink Store. This year's main innovation was the introduction of query rewriting leveraging anchor texts. According to our preliminary performance results, our system performed competivily to other systems. Query rewriting, however, did not lead to substantial performance improvements.

# References


[1] G.V. Cormack, M.D. Smucker, C.L.A. Clarke. Efficient and effective spam filtering and re-ranking for large web datasets. Online at `http://arxiv.org/abs/1004.5168`, 2010.

[2] N. Craswell, D. Fetterly, M. Najork, S. Robertson, E. Yilmaz. Microsoft Research at TREC 2009 – Web and Relevance Feedback Tracks. In *Proc. of the 18th Text Retrieval Conference*, 2009.

[3] N. Craswell, D. Fetterly, M. Najork. Microsoft Research at TREC 2010 Web Tracks. In *Proc. of the 19th Text Retrieval Conference*, 2010.

[4] N. Craswell, M. Szummer. Random walks on the click graph. In *Proc. of the 30th SIGIR Conference*, 2007.

[5] V. Dang, W.B. Croft. Query reformulation using anchor text. In *Proc. of the 3rd Web Search and Data Mining Conference*, 2010.

[6] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proc. of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 668–677, 1998.

[7] R. Lempel and S. Moran. The stochastic approach for link-structure analysis (SALSA) and the TKC effect. *Computer Networks and ISDN Systems*, 33(1–6):387–401, 2000.

[8] M. Najork. The scalable hyperlink store. In *Proc of the 20th ACM Conference on Hypertext and Hypermedia*, pages 89–98, 2009.

[9] M. Najork, S. Gollapudi, and R. Panigrahy. Less is More: Sampling the neighborhood graph makes SALSA better and faster. In *Proc of the 2nd ACM International Conference on Web Search and Data Mining*, pages 242–251, 2009.

[10] Y. Yu, M. Isard, D. Fetterly, M. Budiu, Ú. Erlingsson, P. K. Gunda, J. Currey. DryadLINQ: a system for general-purpose distributed data-parallel computing using a high-level language. In *Proc. of the 8th USENIX Symposium on Operating Systems Design and Implementation*, pages 1–14, 2008.