York University at TREC 2008: Blog Track

Mladen Kovacevic[1], Xiangji Huang[2]
[1]Department of Computer Science & Engineering, York University, Toronto, Ontario, Canada
e-mail: mladen@cse.yorku.ca
[2]School of Information Technology, York University, Toronto, Ontario, Canada
e-mail: jhuang@yorku.ca

Abstract

York University participated in the TREC 2008 Blog track, by introducing two opinion finding features. By initially focusing solely on the sentiment terms found in a document, using two different methods, and not considering the topic relevance, we saw an overall negative impact in the final evaluations. Post-TREC improvements show the necessity of combining opinion weights with the topic relevance scores. A tunable combination function is used, which revealed some interesting findings about the opinion finding features noting the strengths and weaknesses. We also introduce the Compass Information Retrieval system, which is based on Okapi, as the driver by which these experiments were conducted.

1 Introduction

This paper discusses two opinion finding features, sentiment term frequency and sentiment tf-idf term weight. They are implemented by the newly developed Compass Information Retrieval (IR) engine, used in the Blog retrieval task at TREC 2008. Compass IR is a java based application built on the Okapi Basic Search System (BSS), which provides capabilities in preparing data sets, running experiments, and the extendible ability of easily implementing information retrieval techniques. Two baselines and two opinion finding runs were submitted for the TREC 2008 Blog Track. Several more experiments were conducted after the TREC conference deadline, and we make note of those results here as well. By making use of a pre-determined sentiment word list, the opinion finding techniques were used on provided baseline runs to provide a new ranking. As overall results show us, not considering any topical relevance in the re-ranking algorithms is detrimental to the final results. Notable improvements can be seen using combination functions.

2 Background and Motivation

The Compass IR system was concurrently in the process of development and used in preparing the run submissions to TREC 2008 for the Blog track. Okapi BSS is the back-end engine of Compass IR, being integrated for applying the BM25 weighting function [1] to documents, and providing ranked, relevant results. Okapi is an information retrieval system based on probabilistic models of Robertson and Sparck Jones [9], which has been the primary system used in numerous TREC conferences, in various applications and organizations. This includes being used as a platform for contextual information retrieval processes [4], to Chinese text retrieval experiments [2], and early TREC conferences dating back to 1995 [10]. A search term is assigned weight based on its within-document term frequency and query term frequency. The BM25 weighting function applies document weight as shown in equation 1.

$$w = \frac{(k_1+1)*tf}{K+tf} * \log \frac{(r+0.5)/(R-r+0.5)}{(n-r+0.5)/(N-n-R+r+0.5)} * \frac{(k_3+1)*qtf}{k_3+qtf} \quad \oplus \quad k_2 * nq * \frac{(avdl-dl)}{(avdl+dl)} \tag{1}$$

where N is the number of indexed documents in the collection, n is the number of documents containing a specific term, R is the number of documents known to be relevant to a specific topic, r is the number of relevant documents containing the term, tf is within-document term frequency, qtf is within-query term frequency, dl is the length of the document, $avdl$ is the average document length, nq is the number of query terms, the k_is are tuning constants (which depend on the database and possibly on the nature of the queries and are empirically determined), K equals to $k_1 * ((1 - b) + b * dl/avdl)$, and $\oplus$ indicates that its following component is added only once per document, rather than for each term [3].

This was the weighting method we used in developing the primary baseline runs for the Blog track. The opinion finding features presented in this paper did *not* incorporate the BM25 function although we leave this open for future work when using Compass IR with Okapi as the ranking system.

3 Blog Tasks: Baselines to Opinion Rankings

3.1 Overview

There were several tasks associated with the Blog track at TREC 2008. The primary tasks were the opinion finding task and the blog distillation task. Overall, submissions were based on four key tasks.

- Baseline adhoc (blog post) retrieval task
- Opinion finding (blog post) retrieval task
- Polarized opinion finding (blog post) retrieval task
- Blog finding distillation task

This year, we participated in the first two of the above tasks, namely the baseline adhoc and opinion finding retrieval tasks. The Blog collection [5] was the same one initially used in the TREC 2006 Blog track [8], and all of our runs were based solely on permalink documents provided in the data set.

The assessment procedure involved assessors judging the 50 newly added 2008 topics, while the judgments from the previous years, 2006 and 2007, were reused for the first given 100 topics. A total of 150 queries of various topics were provided for the tasks. Pools for judgments were created from participating group run submissions. The documents in the pools were then assigned a value according to the following scale:
0 - not relevant
1 - topic relevant, no opinion
2 - topic relevant, negative opinion
3 - topic relevant, both negative and positive opinion
4 - topic relevant, positive opinion
When the final results given were evaluated, two sets of evaluations were provided for each run. The first, topic relevance, would only measure effectiveness according to all those labelled 1 through 4. Meanwhile, the opinion relevance evaluation would consider only those labelled from 2 to 4. The polarity tasks went even further to determine the differences between the *orientation* of the opinion noted by labels 2, 3, 4 as required.

The goal of the baseline adhoc retrieval task was to provide rankings of blog posts which contained relevant information about the given target, ensuring that all *opinion-finding techniques were turned off*.

The opinion retrieval task re-ranks the results from the baseline adhoc retrieval task, by further placing emphasis on those documents expressing an *opinion* about a given topic target. Being more of a subjective task, the target needed not be the *topic* of the post, rather *opinions about the target* needed to be present.

3.2 Submission Requirements

There were 150 query topics provided by TREC, and a total of 2 topic-relevance baselines were permitted to be submitted. One of which had to be an automatic, title-only run. We submitted runs *york08bb1* and *york08bb2*, both being automatic, title-only runs.

The opinion finding task accepted up to 24 total possible submissions. Four runs were allowed to be based on the two previously submitted baseline runs. TREC also provided 5 standard topic-relevance baseline runs submitted by various groups, for which up to 4 runs were accepted for each baseline. The more submissions that could be evaluated, the more accurate it was in determining the validity of any conclusions. Due to the concurrent development of Compass IR, we were only able to complete two submissions, *york08bo1a* and *york08bo3b*, both being automatic, title-only runs. They were based on re-evaluating baselines 1 and 3 provided by TREC, and used two different opinion finding features.

4 Opinion Retrieval Features

4.1 Experiments

Our experiments were conducted on a single-processor Intel(R) Pentium (R) 4 CPU 3.40GHz server with 1GB of memory. The server ran Fedora 8 Linux distribution, on the 2.6.23 kernel.

Two adhoc retrieval, automatic, title-only runs and two opinion retrieval, automatic, title-only runs were provided. The adhoc retrieval was based on the BM25 weighting function, while the opinion ranking methods incorporated, (A) Sentimental term frequency and (B) Sentimental term tf-idf weight.

4.2 Data Cleaning and Indexing

The permalink documents of the blog posts were given as raw HTML content which needed to be parsed and cleaned into a state which could be indexed by the Okapi BSS. A java based TREC Pre-processor engine was written, as part of the Compass IR package, to perform this work. Integrated into the TREC Preprocessor is the HTML Parser [7], which automatically removed all HTML tags, and provided us with the body text of each document. Since all queries were in English, documents were further cleaned by stripping away any non-ASCII characters, and limiting the final document size at approximately 3MB. A vast majority of documents were under the 100 KB mark, however, there were still a fair amount that were quite large. Because of this, some documents were even left out of the indexing process. The final, cleaned version of the permalink documents amounted to approximately 25GB.

This was a unique data set for Okapi, since in other TREC tracks which used Okapi, although data sets may have been large, on a per-document level, the index entries were normally much smaller [3]. In fact, this large data set required us to break up the data into smaller textual databases, for which a distributed index was then generated. After collecting all the statistics for the associated terms based on this distributed index, we later needed to perform merging to get the final weights and rankings.

Two Okapi indexes were generated from the result set. The first index was used to quickly return the content of a document, given its document (permalink) number. The second index was used for generating ranking document lists, based on the BM25 weight function, for a given TREC query.

4.3 Baseline Adhoc Retrieval

The baseline adhoc retrieval approach we took was applying BM25 on the data set provided by the TREC queries. Each query was read in by Compass IR, and passed to Okapi for ranking. The query terms were extracted, and gathered into a list. Stemming and stop-word removal was applied, and a count of the occurrences of each term is kept. Duplicates were thus removed. We only submitted automatic, title-only runs.

4.4 Opinion Retrieval Techniques

We conducted two key opinion retrieval techniques to re-rank the baseline adhoc retrieval results. They were both simple, and were designed specifically to provide baseline *opinion* retrieval results to which many more techniques can then be compared against. Resource and time constraints prevented us from providing submitting additional runs and implementing the methods we were aiming for. Post-TREC, these baseline runs were in fact improved on, and a tunable combination function was implemented. The combination function did the following:

$$w = a(opinion_{score}) + (1 - a)(topicrel_{score}) \tag{2}$$

where a is a tunable parameter between 0 and 1. The higher the value, the more weight is assigned to the opinion score and the lower the value to the topic relevance score. Also, the $opinion_{score}$ and the $topicrel_{score}$ were the normalized values for the given document over the maximum opinion or topic relevance scores. Thus, for those baselines that do not provide a ranking weight that is based on some scoring method, rather simply go in descending order sequentially by constant value, most likely would not yield beneficial results from using this combination method.

The general approach taken to apply the opinion weight for both the sentiment term frequency as well as the sentiment term tf*idf frequency, can be described as:

Input: a baseline run submission
Output: a re-ranked baseline determined by opinion retrieval algorithm

Method:
(1) Read in the 1000 documents for each query
(2) Apply an opinion ranking algorithm and set new weights for each document
(3) Re-rank the documents according to new weights

Figure 1: General approach to opinion retrieval

For each of the given queries, However, we did not make use of the query term information for the opinion retrieval techniques. Rather, we applied the following method for each document in the baseline .

The two methods of calculating the new document weights were extremely limited because we only incorporated the opinionated feature of the documents, and discarded the topic relevance. This intuitively was not expected to produce the best overall results, but by comparing the two methods, we see drastic differences in overall results as well. By applying the combination function (2), improvements could be seen.

Both approaches make use of detecting sentiment words in the given documents. This was done by using a predetermined, static list of sentiment terms and detecting them during the statistics gathering stage described in Figure-2.

Input: Document number from baseline file
Output: Document with newly calculated weight

Method:
(1) Lookup raw document text from one of distributed databases
(2) Collect statistics about the document, namely
  (3) Number of terms in the document
  (4) Number of sentiment terms found
  (5) Number of target terms from the query
  (6) Positioning information collected of where sentiment/target terms reside positionally
(7) Calculate new document weight given collected stats

Figure 2: Gathering statistics used in determining final document weight

4.4.1 Sentiment term frequency

The first basic weight function was a basic sentiment term frequency function counting the number of sentiment terms found in the document and normalizing the result over the total number of terms in the document.

The first run, *york08bo1a*, used this method against *baseline1* provided by TREC. It was lacking in two ways. First, it did not take into consideration the terms found locally in the document normalized with the *entire* document set. Second, it did not consider any target references in relation to the sentiment terms themselves. Since there were a total of 1000 documents returned per query, and those 1000 are re-ranked blindly according to sentiment word appearance, clearly, those documents that are not relevant but opinionated will potentially be placed higher in the rankings than other relevant (opinionated) documents.

Future work for this method could address the above noted problems by having a way to rank the terms globally considering the entire data set and extending it to make use of sentiment term proximity relative to target terms. Several proximity based methods were used in previous years as well [6]. Our improved approach post-TREC relied on the topic relevance score being incorporated by equation-2, thus the need for involving the query terms themselves was not considered.

4.4.2 Sentiment term tf-idf weight

The second run, *york08bo3b,* was run against *baseline3* provided by TREC. This approach only looked at sentiment terms as well, applying the tf-idf formula to the sentiment terms found in the document, setting the tf-idf weight to be the new score.

The tf-idf weighting function can be defined as follows:

$$weight = \sum_{i=0}^{n} tf_i * idf_i \tag{3}$$

where *tf* is the sentimental *term frequency* in the document and is calculated as,

$$tf_i = \frac{stf_i}{mstf} \tag{4}$$

where *stf* is the frequency of the sentiment word in the document and *mstf* is the maximum frequency of any sentiment word found in the document.

The *idf (inverse document frequency)*, is defined as:

$$idf = \ln \frac{N}{dt} \tag{5}$$

where *N*, is the total number of documents in the collection and *dt* is the number of documents in which the sentimental term is found.

This method extends the first by considering the sentiment terms found over the entire collection. It is still lacking as it does not take into account the target terms from the query whatsoever, nor does it consider the provided topic relevance baseline runs.

5 Results

The two approaches taken in the experiments, *Sentimental term frequency* and *Sentimental term tf-idf weight*, provided interesting results, however, it was unfortunate that we did not provide a direct head-to-head comparison. Instead, we look at the percentage differences that were achieved from the baselines provided compared with the opinion retrieval techniques and compare those.

Two runs were submitted for the baseline runs, *york08bb1* and *york08bb2*. Unfortunately, *york08bb1* provided incorrect results due to incorrect system configuration settings. With the *york08bb2* submission, some system configuration settings were changed, yielding slightly better results, however, the settings had still been incorrect, and do not give a good representation of the system. We repaired these configurations post-TREC and developed a few more notable baseline runs. These runs worked with a single large Okapi index.

Baseline	MAP		R-prec		P@10	
	topicrel	opinion	topicrel	opinion	topicrel	opinion
york08bb1	0.0509	0.0319	0.0973	0.0560	0.1207	0.0660
york08bb2	0.1830	0.1333	0.2660	0.2090	0.5547	0.3733
york08bb3	0.1041	0.0662	0.1622	0.0975	0.1907	0.0960
york08bb4	0.1639	0.1100	0.2500	0.1609	0.2373	0.1320
york08bb5	0.2257	0.1588	0.3118	0.2273	0.4607	0.2793
york08bb6	0.2704	0.1991	0.3463	0.2672	0.5620	0.3713
york08bb7	0.2855	**0.2203**	0.3530	**0.2865**	**0.6053**	**0.4327**
york08bb8	**0.2976**	0.2168	**0.3626**	0.2853	0.5917	0.4138

Table 1: Baseline topic relevance runs using BM25 function

Runs york08bb3 through to york08bb8 adjusted various tuning parameters in Okapi's BM25 algorithm, thus we found york08bb7 and york08bb8 runs to be our top baseline runs based solely on adhoc topic retrieval. It is interesting how across all three measures of the opinion scores for these runs, is that york08bb7 edges out york08bb8. The key difference between these two runs is that york08bb8 takes into account the *adjacency* order of query terms provided in the query. As an example, if "brown fox" was the query, york08bb8 run mandated that "brown" always came before "fox" to determine relevancy. We can see that when looking at the topical relevance, that york08bb8 performs better. We believe that one of the reasons these values were still lower than expected is because our investigation showed that Okapi indeed did not successfully index *all* of the documents.

Two runs were submitted for the opinion finding task, york08bo1a and york08bo3b. Run york08bo1a applied opinion finding feature (A), sentiment term frequency, against baseline1 provided by TREC. Meanwhile york08bo3b applied opinion finding feature (B), sentiment term tf-idf weight opinion finding feature on baseline3 provided by TREC.

Baseline	Run	Average Δ MAP	Max Δ MAP	Min Δ MAP	Med Δ MAP
baseline1	york08bo1a.opinion	-17.47%	268.59%	-98.60%	-7.04%
baseline1	york08bo1a.topicrel	-19.13%	210.60%	-98.46%	-14.45%
baseline3	york08bo3b.opinion	**-62.54%**	**380.00%**	**-99.52%**	**-59.70%**
baseline3	york08bo3b.topicrel	-58.98%	231.76%	-99.25%	-54.25%

Table 2: MAP differences comparing Opinion ranking features vs. adhoc retrieval baselines.

TREC provided results according to topic relevance (runs appended with .topicrel in Table-2) and according to opinion relevance (runs appended with .opinion in Table-2). The average, MAX,

Min and Med Δ MAP represent the MAP changes between the baseline and submitted opinion finding run in order to see the effectiveness of the opinion finding features. Topic relevance was assessed against documents which were relevant, regardless of whether they contained opinions or not. Opinion relevance had to have both topic relevance and opinionated nature in measuring effectiveness. The results indicate the lack of taking the target into account in the re-ranking of the opinionated documents. The worst change in MAP seen for a given query was -99.52%, while the greatest increase in MAP was 380.00%. The greatest average change in MAP over the 150 queries was -62.54%. All three of these largest changes were attributed to the york08bo3b run, which incorporated the sentiment term tf-idf weight algorithm, *solely* for the opinionated terms found in the documents.

The greatest improvement of our opinion finding features of 268.59% and 380% came from queries 920 and 898. These queries are defined as "andrew coyne" and "Business Intelligence Resources" respectively. It is not yet clear as to why these particular queries yielded the best results, and further analysis needs to be performed.

Run	Max/Min	Query	MAP	R-prec	b-Bref	P@10
york08bo1a	max (2008)	1031	0.5666	0.6364	0.6690	1.0000
york08bo1a	min (2008)	1002	0.0074	0.0000	0.0000	0.0000
york08bo3b	max (2008)	1043	0.5621	0.5863	0.6994	0.7000
york08bo3b	min (2008)	1030	0.0018	0.0000	0.0000	0.0000

Table 3: Opinion feature results for best and worst queries.

Table-3 shows us the queries which yielded the best and worst results for our opinion finding features. We make note of what the queries themselves were for these results. Query 1031 was "Sew Fast Sew Easy", query 1002 was "Wikipedia primary source", query 1043 was "A Million Little Pieces" and finally, query 1030 was "System of a Down". We only looked at the best/worst queries from the new 2008 provided queries and not queries from previous years. For each of the queries listed, the results provided equal or lower values from the baseline runs.

Post-TREC results yielded a noticeable improvement over the submitted opinion finding runs. The difference is attributed to using a combination function between the opinion and topic relevance initial ranking, to come up with a new score for the given document. Table-4 shows these results using the term frequency method on baseline1 provided by TREC. The first row shows how the given baseline performed for a baseline comparison.

Baseline	MAP		R-prec		P@10	
	topicrel	opinion	topicrel	opinion	topicrel	opinion
baseline1	**0.3701**	0.2639	0.4156	0.3189	**0.7307**	0.4753
york08bo1a	0.2994	0.2175	0.3738	0.2808	0.4867	0.3433
york08bo2r	0.3429	0.2543	0.4109	0.3177	0.5400	0.4127
york08bo2br	0.3657	**0.2674**	**0.4206**	**0.3239**	0.6680	**0.5000**
york08bo2cr	0.3572	0.2624	0.4187	0.3209	0.6067	0.4653

Table 4: Opinion finding feature results using Term Frequency approach on baseline1 over all 150 queries

We can see by using our combination function, that opinion retrieval was noticeably increased over the original submitted opinion finding run which did not combine the topic and opinion relevance. We can also see how york08bo2br has improved the opinion score over the baesline run itself. Since york08bo2br only assigned 25% weight on the opinion score, it looks like that was enough to give the boost to the opinionated documents to improve performance. Table-6 describes how all the runs are actually tuned and configured.

When re-running the combination function on the sentiment term TF*IDF opinion finding method, we only ran it against baseline1 and so we do not compare the results to the initially submitted runs. Instead we directly compare against the baseline we ran it against.

Baseline	MAP		R-prec		P@10	
	topicrel	opinion	topicrel	opinion	topicrel	opinion
baseline1	**0.3701**	**0.2639**	**0.4156**	**0.3189**	**0.7307**	**0.4753**
york08bo3r	0.2292	0.1493	0.2899	0.1887	0.3613	0.2260
york08bo3ar	0.3094	0.2092	0.3706	0.2684	0.5260	0.3387
york08bo3br	0.2618	0.1734	0.3285	0.2250	0.4120	0.2660

Table 5: Opinion finding feature results using Sentiment Term TF*IDF approach on baseline1 over all 150 queries

It seems evident that the sentiment term TF*IDF approach was not very effective in re-ranking documents from the initial adhoc topic retrieval run. The less weight that was given to the opinion score generated by this opinion finding technique, the better the score. This represents the fact that the baseline score itself was better than this approach. When comparing the two approaches, the simple sentiment term frequency method, in combination with the topic relevance runs, was able to improve on the baseline adhoc opinion retrieval method. Although the improvement was quite small, only about 1.3% improvement, it did not worsen the result like the other methods have.

Opinion Run	**Description**
york08bo1a	Term frequency calculation along with no combination methods
york08bo2r	Term frequency with combination function, using $a = 0.75$
york08bo2br	Term frequency with combination function, using $a = 0.25$
york08bo2cr	Term frequency with combination function, using $a = 0.50$
york08bo3b	Sentiment term TF*IDF with no combination methods
york08bo3r	Sentiment term TF*IDF with combination function, using $a = 0.75$
york08bo3ar	Sentiment term TF*IDF with combination function, using $a = 0.25$
york08bo3br	Sentiment term TF*IDF with combination function, using $a = 0.50$

Table 6: Descriptions of Opinion Finding Features

6 Conclusions

The contributions of our work are as follows. First, we implemented a robust, easy to extend and use information retrieval system, Compass IR, which uses the Okapi Basic Search System as the primary ranking engine. Second, we incorporated several techniques into Compass IR to provide clean baseline runs with which future adhoc retrieval and opinion finding techniques can be implemented. Adhoc retrieval baseline runs submitted were poor, however, later enhancements provided mediocre results using the BM25 weight function of Okapi. The sentiment term TF*IDF opinion retrieval technique degraded the overall opinion ranking results, meanwhile, the simpler sentiment term frequency method improved opinion retrieval in conjunction with a combination score function.

Future work should continue building on the Compass IR infrastructure to incorporate various new research methods. It is important to keep the task requirements clear; to find opinionated documents that are relevant for the given topic. To achieve this, proximity functions can be used to find out how many of the sentiment terms are in close range to the target terms themselves.

7 Acknowledgements

We would like to thank Miao Wen, Damon Sotoudeh-Hosseini, Xiaoshi Yin and Andrew MacFarlane for their support and generous suggestions.

References

[1] M. Beaulieu, M. Gatford, X. Huang, S. Robertson, S. Walker, and P. Williams. Okapi at TREC-5. In *Proceedings of TREC-5*, pages 143–166, 1997.

[2] X. Huang and S. Roberton. Okapi Chinese Text Retrieval Experiments at TREC-6. In *Proceedings of TREC-6*, pages 137–142, 1998.

[3] X. Huang, D. Sotoudeh-Hosseini, H. Rohian, and X. An. York University at TREC 2007: Genomics Track. In E. M. Voorhees and L. P. Buckland, editors, *The Sixteenth Text REtrieval Conference Proceedings (TREC 2007)*, volume 500-274. NIST Special Publication, 2007.

[4] X. Huang, M. Wen, A. An, and Y.-R. Huang. A Platform for Okapi-based Contextual Information Retrieval. In *Proc. of the 29th ACM SIGIR Conference*, 2006.

[5] C. Macdonald and I. Ounis. The TREC Blog06 Collection: Creating and Analysing a Blog Test Collection. In *DCS Technical Report TR-2006-224*. Department of Computer Science, University of Glasgow, 2006.

[6] C. Macdonald and I. Ounis. Overview of the TREC 2007 Blog Track. In E. M. Voorhees and L. P. Buckland, editors, *The Sixteenth Text REtrieval Conference Proceedings (TREC 2007)*, volume 500-274. NIST Special Publication, 2007.

[7] D. Oswald, S. Raha, I. Macfarlane, and D. Walters. HTML Parser. http://htmlparser.sourceforge.net.

[8] I. Ounis, M. de Rijke, C. Macdonald, G. Mishne, and I. Soboroff. Overview of the TREC-2006 Blog Track. In E. M. Voorhees and L. P. Buckland, editors, *The Fifteenth Text REtrieval Conference Proceedings (TREC 2006)*, volume 500-272, page 17. NIST Special Publication, 2006.

[9] S. Robertson and J. Sparck. Relevance Weighting of Search Terms. In *Journal of the American Society for Information Science*, number 27, pages 129–146, May-June 1976.

[10] S. Robertson, S. Walker, M. Beaulieu, A. Gull, and M. Lau. Okapi at TREC-3. In *Proc. of TREC-3*, 1995.