University of Waterloo at TREC 2011: Microblog Track

Adam Roegiest and Gordon V. Cormack
David R. Cheriton School of Computer Science, University of Waterloo

1 Introduction

For the first year of the Microblog Track, a real time ad hoc search task was determined to be a suitable first task. The goal of the track is to return the most recent but also relevant tweets for a user's query. Participating runs will be officially scored using precision at 30. Other experimental scoring measures will be evaluated in parallel to the official measure.

As this was the first year for the Microblog Track, our primary goal was to create a baseline method and then attempt to improve upon the baseline. Since the only task was to perform a real time ad hoc search for the track, we decided that the task would be best suited by using a traditional search methodology. In doing so we used the Wumpus Search Engine[1], which was developed by Stefan Büttcher while at the University of Waterloo.

2 Experimental Setup

2.1 Corpus

From May 27th to June 3rd, the provided HTML crawler[2] was run after which the tweets were repaired using the provided repair code. Once it was learned that the HTML crawler returned tweets with null text bodies, the supplied repair code was modified to refetch tweets with null bodies. The modified repair script was run six times until it was no longer feasible to continue trying to repair tweets. Table 1 shows the distribution of tweets in our copy of the corpus. We note that status code 200 denotes manually posted tweets, 302 denotes Twitter supported retweets, 403 denotes tweets the posting user has protected, and 404 denotes deleted tweets. We note that any tweet marked as 403 or 404 has no associated information as it is no longer available for public use.

[1] http://www.wumpus-search.org
[2] Available at `github.com/lintool/twitter-corpus-tools`

Table 1: Corpus distribution

HTTP Status	Num. of Tweets	Num. null text
200	14,313,888	14
302	1,137,183	25,571
403	138,560	138,560
404	552,181	552,181

No further attempt was made to retrieve the remaining 25,585 tweets as we were unsure as to whether or not the tweets were still available, e.g. not protected or deleted. In addition, we desired to begin working with the corpus and did not believe the missing tweets would significantly affect our results.

2.2 Search Methodology

Our basic idea was to use a wisdom of the crowds approach for our search methodology. To do this for each topics we took 30 sets of search results, created using feature engineering and a variety of ranking methods, and combined them using reciprocal-rank fusion (RRF) [5] as a method of meta-ranking the results. The RRF formula used is as follows

$$RRFscore(t) = \Sigma_i \frac{1}{60 + r_i(t)}$$

where $r_i(t)$ is the rank of the tweet t in result set i.

Feature engineering was performed for each query by creating six different indexes for Wumpus, where the index stores each tweet's features. Table 2 briefly describes the features used for each index. Tweets were ranked using five different methods, which are outlined in Table 3, on each index. These combinations produce our 30 sets of results. Once all the results were combined we took the 30 highest scoring tweets for each and returned those as our relevant tweets.

Table 2: Description of engineered features

Index	Description
1	This index includes stemmed and unstemmed versions of words present in the tweet. The stemming is accomplished using a Porter stemmer. In addition, word bigrams are included in the index.
2	Identical to Index 1 but with no word bigrams present.
3	This index contains only unstemmed versions of words and word bigrams.
4	Idetical to Index 3 but contains no word bigrams.
5	Instead of words as before, character 3-grams are used in the index. Subsequently, stemming is not used as it has no benefit.
6	Identical to Index 5 but uses character 4-grams.

We note that the actual query issued to Wumpus was not just the provided topic but each whitespace delimited word was treated as its own search term. If this was not done then Wumpus would attempt to look for tweets containing exactly the topic phrase and this is not generally a desirable behaviour for a search engine.

2.3 Runs

In this section we describe the official and unofficial runs that we conducted. Runs 1 through 4 were the runs submitted to TREC for official evaluation and so we give the run ID for each of these.

2.3.1 Run 1 - Baseline (waterlooa1)

Our baseline run performs exactly as outlined above with no additions or changes. That is, we use the 5 ranking methods on each of the 6 indexes and use RRF on the 30 sets of results.

2.3.2 Run 2 - HTTP 200 Only (waterlooa2)

This run is identical to our baseline except only those tweets with a HTTP status of 200 are used. The intent with this run was to boost our return of highly relevant tweets as explicit retweets could not be matched.

2.3.3 Run 3 - Query Expansion (waterlooa3)

For this run we decided to perform pseudo-relevance feedback. Using the GOV2 corpus (from the Terabyte Track) as a language model, we performed KLD-based [3] and Okapi-style [2] pseudo-relevance feedback for each query over the top 15 documents and returning 8 feedback terms. Due to some internal limitations, we did not use Cover-Density Ranking for this run. This resulted in a total of 48 result sets being used rather than the standard 30. From our use of RRF, these additional 18 result sets could have improved the final results returned, however, we did not believe any such improvement would be significant.

2.3.4 Run 4 - Recency Blending(waterlooa3)

This run takes the results of the first run and blends the results with respect to how recent the tweet is. The goal here to use the idea that the search request happens when the topic is (relatively) popular and so more recent tweets are more likely to be relevant to the topic. Namely, each tweet, t, is re-scored as follows:

$$recscore(t) = \left(\frac{\text{tweetid}(t)}{\text{max-tweetid}}\right) * \text{run1-score}(t)$$

where max-tweetid is the upper most recent tweet ID by the time of the query.

2.3.5 Run 5 - Conjunctive Query

This run modifies the baseline by creating a new set of query terms by taking the pairwise conjunction of the original query terms. The intent here is to return documents which contain more of the original query terms.

2.3.6 Run 6 - Stopword Removal

This run removes all stopwords from the query terms (excepting those that are part of quoted terms) as a means to remove the possibility of Wumpus returning tweets that are relevant to stopwords. Our goal in doing so was to hopefully improve the quality of the search results.

2.3.7 Run 7 - Tweet-based Query Expansion

This run builds upon the theory of run 3 by using the all the tweets up until the oldest query tweet time (e.g. the earliest query) and builds a Wumpus-style language model using those tweets. The intent here is to see how much effect the language model has on the results returned.

Table 3: Ranking Methods used

Method	Description
Okapi BM25[6]	BM25 is a bag-of-words retrieval method that ranks documents based upon how often query terms appear in the document and does not take into account any relationships between query terms in the document (e.g. proximity).
Ponte-Croft Language Modelling[7]	This method infers a nonparametric language model for each document and then ranks each document based on the query. That is, a document is ranked for a query using the product of the probability of producing query terms in the document and the probability of not producing other terms.
Language Modelling with Dirichlet Priors[8]	A language model, in this method, is treated as a multinomial distribution and the Dirichlet distribution is used to smooth the probability of word occurrence.
Cover-Density Ranking[4]	Given a set of query terms Q1, ..., Qn, this method builds a boolean AND for all subsets (e.g. Q1^Q2^Q9^Qn) and ranks these subsets by the sum of their terms' Inverse Document Frequency values. Then all documents are ranked based upon the largest subset they contain.
Divergence from Randomness[1]	This method uses inverse term frequency, the ratio of two Bernoulli processes, and the hypothesis that the term frequency density is inversely related to the length to create a ranking formula for documents.

2.3.8 Run 8 - Conjunctive Query and Stopword Removal

In an effort to boost the effectiveness of runs 5 and 6, we removed all stopwords before taking the pairwise conjunction as this would remove silly conjunctions, e.g. '(at^the)'.

2.3.9 Run 9 and Run 10 - Query Expansion using Okapi-style feedback only

These two runs redo runs 3 and 7, respectively, but remove the KLD-based queries, meaning that only 24 sets of results are used in the computation of the final ranking. The purpose of this run is to see how much effect was produced by using KLD queries, e.g. to determine how much affect the 18 additional queries had due to RRF.

2.4 Evaluation

Tweets were judged to be relevant by NIST assessors after the results from all participating teams were pooled. Assessors were asked to judge tweets as relevant and highly relevant with regards to the topic. In addition, only tweets that primarily contained the English language were allowed to be judged relevant. According to the assessment of returned tweets, 33 topics were found to have highly relevant tweets and 49 topics were found to have relevant tweets. We did not augment the set of relevant tweets with any new tweets found by our unofficial runs. Runs were scored based upon precision at 30 for each topic.

3 Results and Discussion

Tables 4 and 5 provide summary results for topics with highly relevant tweets only and topics with relevant tweets[3] for posterity. In addition, our results did not differ regardless of which set of qrels were used. From these summary results, our baseline run appears to have performed most of the time at or above median, which may indicate that it is a good baseline. Our second run performed not as well as hoped, however, we feel that this was due to the relative paucity of highly relevant tweets, which amounts to about a third of the tweets judged to be relevant. The improvement in run 3 was expected due to the nature of query expansion. We found that while recency blending did not extremely improve results over the baseline it did tend to improve the results and would seem to indicate that relevant tweets tend to

[3]We note that in the TREC notebook version, the presented results for Run 7 for highly relevant tweets were incorrect

occur closer to the query time as we predicted.

We first consider whether any of the official runs improves substantially over the baseline. Run 3 appears to yield a significant improvement ($p = 0.016$) when only highly relevant tweets are scored, and also ($p = 0.00004$) when all relevant tweets are scored. Since three hypotheses were tested at once, we apply Bonferroni correction, and find that the both improvements are still significant ($p = 0.048$ and $p = 0.00012$, respectively).

Of the unofficial runs, Run 7 and Run 10 improve significantly on the baseline, after Bonferroni correction, both when highly relevant tweets and all tweets are considered. Run 9 appears also to improve on baseline; however, it is arguable whether the improvement when highly relevant tweets are scored is significant, after correcting for multiple hypothesis testing. According to a strict application of Bonferroni (which is known to be conservative) it is not.

The results would seem to indicate that both run 5 and 6 did not significantly improve upon the baseline. This unfortunately extends to run 8, which was the combination of methodologies for runs 5 and 6. Run 8 appears to have marginally improved upon run 6 with respect to all relevant tweets but has decreased performance when compared to run 5. Similarly, while run 8 appears to have returned more highly relevant tweets than either of its component runs it would appear that it did so at the expense of returning tweets at or above the median.

Our results would seem to indicate that the language model used for query expansion does matter. Namely, our tweet based language model for query expansion still does quite a bit better than our baseline and still appears to give some improvement over the initial query expansion run. This would indicate that a better tweet-based language model, e.g. consisting only of English tweets or larger size, may further improve retrieval results.

We noticed a strange phenomena when examining our summary tables. It appears that for all non-query expansion runs the percentage of topics that have relevant tweets returned at or above median is higher for highly relevant tweets than all relevant tweets. But this appears to be the opposite for all query expansion runs. Unfortunately, we have no good explanation as to why this is happening.

Furthermore, runs 9 and 10 would seem to indicate that the additional results provided by the KLD-based pseudo-relevance feedback did not seem to significantly affect the results. There is a minor drop in the average number of relevant results returned @ 30 but not nearly dramatic enough to indicate that the extra results significantly boosted the results. What is interesting is that run 9 has a higher percentage of topics with relevant tweets returned at or above the media than does run 3 but this appears to be the opposite for runs 7 and 10. We were unable to determine a reasonable explanation for this fact.

In an examination of the relevant tweets we came to a discomforting realization; that 12 of the relevant tweets were marked as HTTP status 403/404. As we had collected the data set fairly soon after it was released we had hoped that this would not happen. While it is likely that missing these 12 tweets did not terribly affect our performance, it is possible, however, that teams who collected the data set after we did would be penalized for doing so. Thus, a true comparison of performance may require that any HTTP status 403 or 404 tweets from any group be removed from the relevant set of tweets. But this could substantially affect the number of relevant tweets, which at this time seems rather low. In any future iterations of this track, it would be desirable if a stable data set could be used or enforced.

4 Future Work

From the performance of run 7, we would like to investigate the utility of a better tweet-based language model. For example, we would like to use a tweet-based language model composed of English tweets only as we believe this will correspond better to the provided topics. In addition, we would like to use a dynamic language model, by which we mean a language model that scales to each topic's most recent tweet as opposed to our current method of using the oldest topic's most recent tweet as our cut-off. Finally, we would also like to investigate an idealized run wherein we use the whole corpus (or some other sizable collection of tweets) as the language model.

Due to the increasing usage of hashtags as part of the Twitter experience, we initially thought to use hashtags as a query method. However, initial testing on the example topics showed an inconsistent pattern of usage, e.g. some topics had very high usage and some had little to no usage. Thus, we ruled out doing this automatically on the official topics. In examining the set of relevant tweets, we saw a similar pattern where some topics had decent hashtag usage and others did not. While this still prohibits an automatic search system, we will in the future conduct a manual run using hand selected hashtag(s) from those topics with adequate hashtag usage.

We had initially planned to crawl the URLs contained in tweets and use the first 10,000 bytes as an additional source of information, however, we were

unable to accomplish this. As around 82% of the relevant tweets contain at least one URL, we feel that such an endeavour may have a non-trivial effect on the retrieval of tweets. Accordingly, if the track continues next year we will endeavour to use the linked content as an additional source of information.

5 Conclusions

This year the University of Waterloo created a baseline for our own measurement. While not all of our modifications to this baseline proved to be beneficial we did see some improvement over the baseline using various techniques. The best performing of these techniques appears to be query expansion, especially when a tweet-based language model is used. In addition, we performed well on a majority of topics and even achieved the highest number relevant documents returned @ 30 on a few topics.

References

[1] Gianni Amati and Cornelis Joost Van Rijsbergen. Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Trans. Inf. Syst.*, 20:357–389, October 2002.

[2] Bodo Billerbeck and Justin Zobel. Questioning query expansion: an examination of behaviour and parameters. In *Proceedings of the 15th Australasian database conference - Volume 27*, ADC '04, pages 69–76, 2004.

[3] Claudio Carpineto, Renato de Mori, Giovanni Romano, and Brigitte Bigi. An information-theoretic approach to automatic query expansion. *ACM Trans. Inf. Syst.*, 19:1–27, January 2001.

[4] Charles L. A. Clarke, Gordon V. Cormack, and Elizabeth A. Tudhope. Relevance ranking for one to three term queries, 2000.

[5] Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '09, pages 758–759, 2009.

[6] K. Sparck Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments. In *Information Processing and Management*, pages 779–840, 2000.

[7] Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '98, pages 275–281, 1998.

[8] Chengxiang Zhai and John Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 334–342, 2001.

Table 4: Results for highly relevant tweets only

	Official Runs				Unofficial Runs					
	1	2	3	4	5	6	7	8	9	10
Percent above median	54.55	51.52	66.67	54.55	57.58	57.58	66.67	60.61	63.64	69.70
Percent at median	36.36	39.39	18.18	36.36	33.33	33.33	21.21	27.27	24.24	15.15
Avg. rel. ret. 30	3	2.88	3.45	3.03	3.09	3.09	3.82	3.30	3.45	3.73
p-value	-	1.00	0.016	0.5	0.34	0.19	0.00005	0.055	0.013	0.003

Table 5: Results for all relevant tweets

	Official Runs				Unofficial Runs					
	1	2	3	4	5	6	7	8	9	10
Percent above median	73.47	73.47	77.55	77.55	73.47	73.47	83.67	73.47	83.67	85.71
Percent at median	12.24	12.24	16.33	12.24	14.29	12.24	12.24	14.29	12.24	8.16
Avg. rel. ret. 30	11.10	10.82	12.29	11.27	11.18	11.10	12.65	11.16	12.22	12.55
p-value	-	1.00	0.00004	0.1	0.4	0.6	0.0000002	0.4	0.00004	0.0000006