Simple Rank-Based Filtering for Microblog Retrieval: Implications for Evaluation and Test Collections

Ben Carterette, Naveen Kumar, Ashwani Rao, Dongqing Zhu
University of Delaware
{carteret,nkumar,ashwani,zhu}@udel.edu

1. INTRODUCTION

The IR Lab at the University of Delaware participated in the first year of the TREC Microblog track. We submitted two runs using two different indexers and ranking functions, one of which filtered results by a score threshold. The results inspired some post hoc analysis of the test collection, which is the main focus of this paper.

We summarize our results as follows:

1. a standard language modeling scoring function combined with a rank-based threshold performs well (above median TREC performance by precision at 30 for the majority of topics);
2. a standard vector space scoring function with no rank-based threshold performs poorly (below median TREC performance by P30 for the majority of topics).

However the two systems also used different language filtering approaches and as such cannot be compared directly.

Post-submission experiments suggest that the major factor influencing results is the use of a rank-based threshold. In particular, thresholding a simple query-similarity score ranking (which has also been filtered for English language results) at rank 30 gives the best overall results by the track evaluation, even though it results in only retrieving tweets that occurred days (or even weeks) before the original query. Since this seems to be "gaming" the evaluation and is clearly not in the spirit of the track, we did some analysis of the test collection to try to determine why this was the case. The majority of this paper is dedicated to that analysis. We summarize the outcomes as follows:

1. tweet relevance is negatively correlated to recency for the majority of topics, query similarity score is negatively correlated to recency for almost all topics, and tweet relevance and query similarity score are positively correlated for the majority of topics;
2. which combined mean that ignoring recency and just thresholding a similarity score ranking will give good results by time-ordered P30 and other measures, though the retrieved tweets occur days before the query;
3. and for the topics for which relevance and recency are positively correlated while relevance and similarity score are negatively correlated, we perform poorly but there are not enough relevant documents to hurt overall performance to a substantial degree.

.

2. MICROBLOG CORPUS

We crawled the corpus in late June of 2011. In our initial crawl we found:

13,689,304	tweets with status code 200
1,052,593	with status code 302
1,399,915	null

For ease of processing, we converted the crawled corpus into the standard TREC document format, with tweet ID as the DOCNO and the tweet text as the document TEXT. However our method for this resulted in some data being lost. The final collection for indexing and retrieval had:

13,654,440	tweets with status code 200
1,050,399	with status code 302
1,399,915	null

We did not attempt to model the real-time search task by excluding tweets that occurred after the query time from the scoring. It is not clear to us how this would affect our final results.

3. SUBMITTED RUNS

One run was generated by the indri[1] system from an index of our entire corpus. We used indri's inference network/language model with pseudo-relevance feedback for scoring. Also, we let the system search only on the 'text' field of the transformed tweets by imposing a field restriction on the indri queries, e.g. #combine(NASA.(text)), where 'NASA' is the query term (not doing this had a strong negative effect on the final evaluation). For pseudo-relevance feedback, we took the 20 highest-weighted terms from the top 15 tweets in the initial ranking and assigned a weight of 0.8 to the original query terms. After indri retrieved the top 1000 tweets for each topic, we further removed tweets that appeared after the query time. We then dropped non-English tweets using Microsoft's language detection API[2]. The final step was to cut the ranking off at the top 6% line for each topic.

The other run used the Lucene[3] system with an index of a corpus pre-filtered for English. We used the Enchant python libraries to filter out non-English tweets, taking a very simple approach of declaring a tweet to be English if more than 30% of tokens in the tweets can be recognized as English. We also used some regular expression preprocessing on the tokens to filter out some common types of tokens known in the tweet world. For example we did not perform English language checks on tokens starting with the '@' symbol or

[1]http://lemurproject.org/indri/
[2]http://msdn.microsoft.com/en-us/library/ff512411.aspx
[3]http://lucene.apache.org

run	eval	P30	MAP	R-prec
udelIndri	allrel	0.308	0.125	0.195
	hirel	0.087	0.063	0.098
- RTs	allrel	0.392	0.157	0.212
udelLucene	allrel	0.086	0.047	0.077
	hirel	0.042	0.040	0.058

Table 1: Official results, and results after filtering RTs (tweets with status 302).

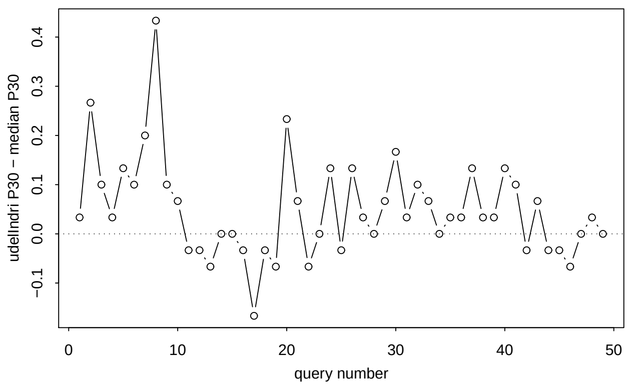

Figure 1: Difference between udelIndri P30 and median TREC P30 for all 49 topics. Our system is at or above the median for the majority of topics.

the '#' symbol or 'http'. After filtering, 6,332,164 tweets remained for indexing. We then used Lucene's default scoring to get our top results for the queries and filtered out the tweets from the result which occured after the query time.

3.1 Results

Mean performance of our two submitted runs on the three primary evaluation measures is shown in Table 1. The indri system performed relatively well, while the Lucene system performed relatively poorly. At the time of submission we did not realize that tweets with status 302 would not be judged; if we had filtered those, our results would have been substantially better.

Despite that, the submitted indri system also performed well relative to median TREC performance, as shown in Figure 1. Only 13 topics are below the median, with another 7 roughly at the median.

However, retrieved results are not "recent" relative to query time and the overall timeline represented in the corpus. As Figure 2 shows, 82% of our retrieved results came from more than a day before the query time (measured using the actual tweet time minus the actual query time), and nearly 40% from more than a week before the query time.

Thus while we performed well by the official evaluation, we feel we should consider the system a failure at the real-time search task.

4. ANALYSIS

In this section we analyze the properties of the test collection that allowed us to do well despite not actually attempting to solve real-time search.

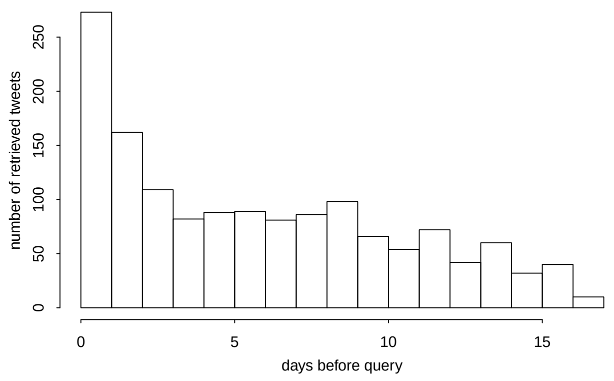

Figure 2: Number of retrieved tweets by our udelIndri system on each day prior to the actual time of the query.

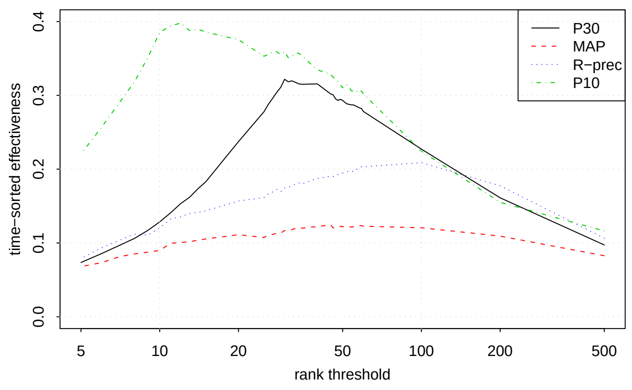

Figure 3: Effect on time-sorted evaluation measures of filtering score-ordered results at different rank thresholds. The best P30 is obtained by thresholding at exactly rank 30.

4.1 Analysis of rank-based filtering

We evaluated a pre-filtered udelIndri that retrieved as many as 1000 results for every topic, and an alternate filtering that just cut the ranking off after rank 30 for all topics. The former run has a time-sorted P30 of only 0.075, while the latter outperforms our original submitted run with a time-sorted P30 of 0.323. In fact, P30 is optimized at exactly rank 30 (Fig. 3), which is surely not a coincidence.

But this is not just a warning against time-ordered P30. As Figure 3 shows, the same results hold true for precisions at other cutoffs (e.g. the best P10 is achieved by thresholding near rank 10; the best P50 is achieved by thresholding near rank 50) and for other measures (e.g. the best R-precision is achieved by thresholding each query at its number of relevant documents). For all measures but MAP, the effect is a very strong one, and even in the case of MAP it is strong enough to overcome the usually-detrimental effect on MAP of thresholding the ranking early.

A possible danger is that it in training a system to any measure, this thresholding effect may be strong enough to overcome any effect from features. Rather than learn fea-

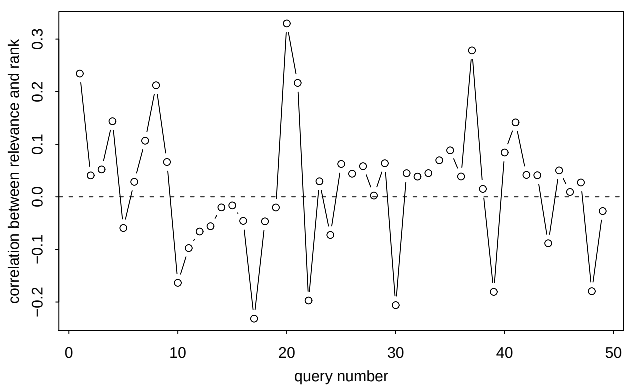

Figure 4: Correlation between tweet relevance and tweet rank when sorted in decreasing order of ID. Negative correlation means recency is more important; 18 topics show a negative correlation.

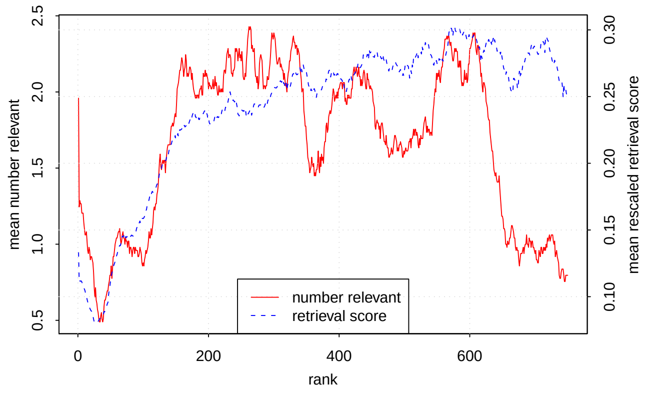

Figure 5: Running counts of relevant documents and running averages of retrieval scores when judged tweets are sorted in decreasing order by ID, averaged over all 49 topics.

ture weights, it seems likely that systems will be trained to threshold at the appropriate rank for whatever the objective function/evaluation measure is.[4]

4.2 Correlation analysis

The analysis in the previous section suggest that a simple rank thresholding of a similarity score ranking can optimize many evaluation measures. Since this is not in the spirit of the real-time search task, we analyzed the test collection in more depth to try to understand why this is the case. The analysis in this section is based mostly on the topics and the judged tweets along with language modeling scores as computed by indri.

First we investigated correlations between recency, relevance, and retrieval score. Figure 4 shows, for each query, the rank correlation between tweet relevance and tweet rank when the judged tweets are sorted in decreasing order by ID. Negative correlation means that more relevant tweets occur deeper in this ranking, or intuitively that recency is not a factor for the topic. Positive correlation means the opposite: more recent tweets are more likely to be relevant, and therefore recency "matters" for the topic.

By this analysis there are only 18 queries for which recency matters and 31 for which it does not. For these 31, a system is likely to be penalized for showing any preference for more recent tweets. Note that there is a correlation between topics for which our simple score thresholding system underperformed the median (shown in Figure 1) and topics for which recency is more correlated with relevance. In particular, we underperformed for topics 11, 12, 13, 15, 16, 17, 18, 19, 22, 44, and 49, eleven of the 18 topics (61%) for which recency is important.

We next computed a retrieval model score for every judged tweet using indri. The result is not shown, but it suffices to say that for all but one topic (topic 13), retrieval score is positively correlated with appearance deeper in the ranking. This means the model tends to assign higher probability of relevance to tweets that occur further before query time.

[4]In the case of R-precision this has the interesting possibility of leading to new ways to predict the number of relevant documents, but that is surely an unintentional side effect.

If recency does not matter for most topics (or is even a negative indicator of relevance), such a system is expected to be successful, despite not serving the RTS task.

This correlation analysis suggests that the collection will generally favor systems that rank for likelihood of relevance and then use a harsh threshold over systems that rank for recency and relevance in some combination.

The precise shape of the correlation matters as well. Figure 5 shows running averages for relevance and retrieval score going down the ranked list of judged tweets (i.e. back in time from the time of the query). For each query, at each rank we count the number of relevant tweets occurring in the next 25 ranks, or take the average retrieval score (linearly rescaled to the range $[0, 1]$) over the next 25 ranks. We then average these counts or means at each rank across 49 topics. Using the next 25 ranks is a running average approach meant to smooth the curves.

Here we can see the shape the negative correlation between relevance and recency takes: there are some relevant tweets that occur right before query time, but then a big drop-off, after which it takes until nearly rank 200 to achieve the same quantity on average. After that relevance tends to be steady for a while (with some peaks and valleys) before dropping off substantially again after rank 600. Meanwhile, query similarity score trends upward over time with only minor fluctuations until after rank 600. Nevertheless, it can be seen on this plot that taking the top 30 highest-scoring tweets will correlate quite well with the parts of the timeline in which relevant documents are most likely to be found.[5]

4.3 Even more analysis

We have simplified some things for the purpose of the analysis above, but those simplifications may affect the conclusions. First, tweet ID may not be well-correlated with time before query; e.g. rank 200 might mean different time differentials for different queries, and thus different conclusions about relationships between recency and relevance. Second,

[5]Not shown: when highly-relevant are counted double, it turns out that the peak around rank 600 is even higher, suggesting that highly-relevant tweets are occurring deeper in the ranking than relevant tweets on average.

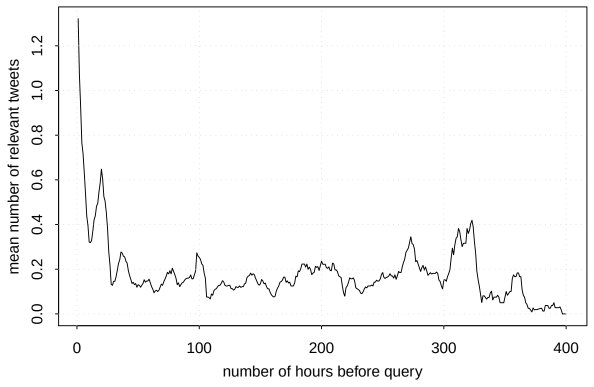

Figure 6: Mean number of relevant tweets in a sliding window of 8 hours (averaged over 49 topics) at each hour prior to the query. Contra Fig. 5, this shows more relevant tweets closer to the query time.

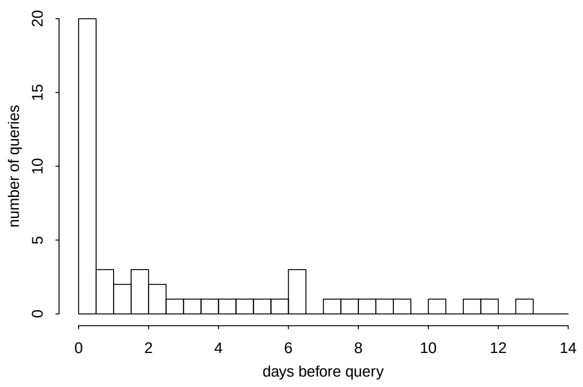

Figure 7: For each 12-hour block, the number of queries with a plurality of relevant documents in that block. This suggests that after the first 12 hours, relevant tweets are roughly uniformly distributed over the timeline relative to query time.

there may simply be more judged tweets from times closer to the query (possibly because of the nature of the TREC submissions[6]), leading mean score to be low in that part of the timeline even though there are relevant tweets. Third, because retweets and non-English tweets were automatically assumed nonrelevant (and thus not present in the qrels), there would actually be more nonrelevant tweets than used in this analysis. If these are not uniformly distributed over time, it would affect the shape of the curve.

Without looking at all submitted runs it is difficult to investigate the second and third points, but they have implications for the procedure used to collect relevance judgments. It may be true that there are many more recent tweets than non-recent tweets due to the time sorting, but it may nevertheless be beneficial to have judgments evenly distributed over the timeline, especially if the negative correlation between recency and relevance is "real".

For the first point, it turns out that there is indeed a discrepancy between tweet ID and actual time of tweet. Figure 6 shows the average number of relevant tweets per hour before the query (using the actual real-world time of the tweet relative to the actual query time). Here it seems that recency *does* matter; about 30% of relevant tweets appear in the first day prior to the query, and more than 20% in the first 12 hours. Why then do these relevant tweets not show up en masse until deep in the ranking by tweet ID? Even if the first 250 ranks occur in the first 12 hours, we would expect relevant tweets to be more evenly distributed over those ranks rather than mostly occurring at the end as Figure 5 shows.

One possibility is that tweet ID and actual tweet time are not well-correlated, but this is easily dismissed by plotting ID against time—the relationship is perfectly linear apart from one or two outliers.

An alternate possibility has to do with the distribution of relevant documents among topics and in the timeline. It may be that there are fewer recent relevant documents than non-recent relevant documents, but recent relevant documents tend to have overlapping time distributions across queries while non-recent relevant documents do not. This would explain why the time-vs-relevance curve starts out high, then drops to a relatively constant number.

In fact, the 18 queries for which recency matters (according to the correlation analysis) make up 37% of the topic set, but only have 27% of all the relevant documents; of these relevant documents, 39% occur in the first 12 hours before the query and 63% in the first 24 hours. This accounts for 47% of all relevant documents for all topics occurring in the first 12 hours and 53% of all relevant documents occurring in the first 24 hours.

But for the other topics, there is little overlap in when the relevant documents appear relative to the query. In Figure 7, we split the timeline into 12-hour blocks and, for each query, found the block containing the plurality of its relevant documents. The figure shows the number of queries that have a plurality of relevant documents in the corresponding 12-hour block. While there is significant overlap in the first 12 hours, after that there is almost no overlap at all. Each query is focused on a different block of time *relative to* the original query. (There may still be overlap in absolute time.)

In other words, those queries that do have many recent relevant results dominate relevance in the first 24 hours, while queries with few recent relevant results tend to have their relevant results clustered within a timespan that is not shared (to any significant extent) by any other query. Yet the former group makes up a minority of both the topic set and the relevance judgments. This may be why retrieval scores averaged across queries are high for the deeper ranks when sorting by tweet ID.

4.4 Summary

We demonstrated that correlations between recency, relevance, and retrieval score conspire to make a simple rank-thresholding system perform well. We further showed that this is likely due to the distribution of relevant tweets among topics and in the relative timeline for each topic.

This analysis is suggestive; it does not prove anything. Nevertheless, it seems to us likely that the hard recency

[6]Based on reading the track mailing list, it seems likely to us that many participants submitted long ranked lists of tweets without realizing they would be re-sorted by recency.

constraint imposed in the evaluation means that automatic systems trained to this corpus will *not* find much gain from using features related to recency. For the P30 evaluation, the best strategy appears to be simply thresholding a ranking based on a relevance/retrieval model at rank 30. This is likely not in the best interest of any user this test collection may be meant to model (except maybe for one that always wants exactly 30 results for a query and doesn't care much about recency).

Furthermore, it suggests some directions for future microblog evaluations: first, ensure that there is significant overlap in the relative times at which relevant tweets appear across topics. This could be accomplished by ensuring that a substantial proportion of the relevant tweets occur very recently (which would also help better model the real-time search task). Second, explicitly penalize systems for retrieving less-recent tweets. The current official evaluation only implicitly penalizes by recency; as we showed above, we avoided this penalization by only retrieving the top 30 highest-scoring tweets, even though those tended to appear on the order of a week prior to the query. Third, try to ensure an even distribution of judgments across the relative timeline; do not focus judging on the most recent blocks of time. (The fact that judging was focused there is likely due to the types of systems participants submitted.)

5. CONCLUSION

While our indri run performed well, we nevertheless consider it a *failure* at performing the real-time search task: its retrieved results, despite being relevant, are not recent relative to query time. This seems to be because of a combination of the distribution of relevant tweets over time in the collection and the official evaluation using only implicit recency penalization after imposing a hard explicit recency constraint on every system regardless of retrieval algorithm.