NUSIS at TREC 2011 Microblog Track: Refining Query Results with Hashtags

Hadi Amiri[1,§], Yang Bao[2,§], Anqi Cui[3,§,*], Anindya Datta[2,§], Fang Fang[2,§], Xiaoying Xu[2,§]

[1] Department of Computer Science, School of Computing, National University of Singapore, Singapore 117543. E-mail: hadi@nus.edu.sg
[2] Department of Information Systems, School of Computing, National University of Singapore, Singapore 117543. E-mails: {baoyang, datta, fangfang, xu1987}@comp.nus.edu.sg
[3] State Key Lab of Intelligent Technology & Systems, Tsinghua National Lab for Information Science & Technology, Dept. of Computer Science & Technology, Tsinghua University, Beijing, 100084, China. E-mail: cuianqi@gmail.com

[§] All authors contribute equally to the work, and are listed in alphabetical order of their surnames.

Abstract: In this paper, we describe our submission to the TREC 2011 Microblog track. We first use URLs as a clue to discover and remove the spam tweets. Then we use both Lucene and Indri to generate a ranked list of results for each query, together with their relevance scores. After that, we use the scores to find out useful hashtags relevant to the query, therefore some previously lower-ranked tweets can be discovered and are re-ranked higher. Query reformulation is considered in two of the four runs in our submissions.

1 Introduction

The NUSIS team participated in the Realtime Adhoc task component of the TREC 2011 Microblog track. The team comprises members from the National University of Singapore and Tsinghua University. We analyzed the task and evaluation methods carefully, and submitted results based on our best understanding.

The task required at least one run with no external and future evidence involved. To achieve this, we first filtered out the "past" tweets according to the timestamp of each of the 50 query topics. Then we ran our algorithm 50 times, generating results for these topics. We provide a broad overview of our approach.

We start off by removing spam tweets. Tweets are usually in short texts; hence most spam tweets contain URLs of their own websites, so that users will visit the "den" from this link. Therefore, presence of URLs in a tweet provides strong evidence of it being spam. We find that many popular URLs are spam URLs [1], and conduct a simple method to remove these tweets.

We constructed indices with Lucene [2] and Indri (for query reformulation) [3], obtaining a relevance score of each tweet in the process. Instead of directly submitting the first 30 results, we adopt an algorithm that generates modified scores for these tweets, which, in turn, re-ranks the current list. Since some tweets may be ranked higher after this modification, the submitted results can be different. In the end, the tweets are sorted by these refined scores (for the adhoc evaluation) or by a combination of both their time and their modified scores (for the balanced evaluation).

The basic idea is to discover relevant hashtags for a query topic. We believe that, when generating

[*]This work has been done by the support of Tsinghua-NUS NExT Search Center.

results with traditional text information, the hashtags mentioned in these tweets are also of interest. Therefore, other tweets containing these hashtags but less relevant texts are also relevant to this topic. In this way, more tweets of interest are discovered.

2 Dataset and Preprocessing

The tweet dataset is downloaded using the twitter-corpus-download-tool [4]. A total of 13,660,436 tweets are downloaded with status code 200. Tweets with status code 302 (1,093,549 tweets), as well as tweets with code 200 but start with "RT", are retweets, which are defined as non-relevant, thus are removed. Some other tweets (1,378,120 tweets) are removed (with code 404) or are protected (with code 403), preventing us from accessing them.

According to the task description, all non-English tweets are considered as non-relevant. Similar as the preprocessing step in a previous work [5], we remove these tweets based on the characters in the text, i.e. tweets containing characters other than Basic Latin and symbols are removed.

Many tweets are too short which result in little information. Tweets with less than five words are discarded.

As mentioned before, tweets with spam URLs are removed. The spam URLs are determined from the following two aspects:

(1) For each URL u which appears more than five times in the corpus, it is considered as a spam URL if $n_u / N_u \leq 0.4$, where n_u is the number of unique users who have posted tweets containing u, and N_u is the total number of times u has appeared.

(2) We also examine the dataset and find some popular domains with high occurrences. Therefore, we manually identified a set of spam domain including: "tinychat", "twittascope", "twitcam", and "twitcast". All URLs in these spam domain are considered as spam URLs.

3 Indexing and Query Reformulations

The indices and relevance scores are generated by Lucene [2] and Indri [3] retrieval systems. We use the default implementation of the popular vector-space model in Lucene. Indri provides a robust query language that both accept keyword queries and complex queries. This language model provides many features like complex phrase matching, weighted expressions, and Boolean filtering, etc. We utilize these features to formulate the queries for the task. In particular for each query we first extract its N-Grams (N = 2, 3). We then construct some weighted expressions using the query and its N-grams. The reason that we use weighted expressions is because of the fact that it allows controlling the impact of each expression (e.g. by varying weights).

Let the query $q = t_1, t_2, \ldots, t_n$, where t_i indicates the i-th term of the query. Using the Indri's query language we construct the following sub-queries:

(1) The whole query with weight 2.0
(2) Its 3-Grams with weight 1.5
(3) Its 2-Grams with weight 1.0, and
(4) All the query terms with weight 0.5.

The following query is then obtained:

Weight (2.0 #*M* (t_1 t_2 … t_n)
1.5 #*N* (t_1 t_2 t_3) 1.5 #*N* (t_2 t_3 t_4) … 1.5 #*N* (t_{n-2} t_{n-1} t_n)
1.0 #*N* (t_1 t_2) 1.0 #*N*(t_2 t_3) … 1.0 #*N* (t_{n-1} t_n)
0.5 t_1 0.5 t_2 … 0.5 t_n
),

where #*X* indicates that the terms should be in order with the max distance of *X*-1. We set *M* and *N* to 30 and 5 respectively in our experiments. We then perform the retrieval using the above reformulations.

4 Refining with Hashtags

The hashtag (word starts with a hash symbol #) is a specific feature in tweets. They usually denote the category or topic of the tweets, to provide a useful annotation. Hashtags are provided by the original author of the tweet; we believe them much relevant to the content, thus can be used to improve the query results.

Formally, let the results without refining be $\{r_n\}$, where each r_i consists of the tweet t_i and its relevance score s_i. Let $\{h_{i,m}\}$ be the hashtags in t_i.

Since the first 30 results are more important in the evaluation, we scan the first 30 tweets to assign scores to the hashtags. For each of these tweets, denote s_i as the score of the hastag $h_{i,m}$. Therefore, hashtags that appear earlier in the result list (in the tweet ranked higher) have higher scores. Note a same hashtag may occur more than once in the tweets (or even in the tweet t_i itself); the hashtag will be assigned scores multiple times.

Then, for each unique hashtag h_k, we add up all the scores it has been assigned. Hence, hashtags that occur more times will have a higher final score, namely $S(h_k)$.

The final score of a tweet is combined with both the original relevance score s_i and the scores from the hashtags it contains, $\Sigma_k S(h_{i,k})$, controlled by two weight factors w_1 and w_2:

$$\text{finalscore}(t_i) = w_1 \cdot s_i + (1 - w_1) \cdot \Sigma_k S(h_{i,k}) + w_2 \cdot s_i \cdot \Sigma_k S(h_{i,k})$$

In practice, we set w_1= 0.85 and w_2= 0.07.

Finally, all the retrieved tweets (mostly more than 30) are re-ranked by the final score. Then the first 30 results are submitted for the first evaluation.

For the balanced evaluation, we consider both the time a tweet is created and its refined final score. We sort the result list by the time and score separately, thus for each tweet we have two ranks. Then we add the two ranks together to get a new index for the tweet, and sort the list again by this index. Finally, the first 30 results from this newly sorted list are submitted.

5 Results and Discussion

According to the judgments, the evaluation includes scores from all 49 topics (topic 50 is dropped)

and from 33 topics which have highly relevant tweets. The primary measure is P@30.

We submit four runs, two with Lucene indexing and two with Indri (query reformulation). For each indexing type, we submit a relevance result and a balanced result.

Our performances are shown in Fig.1 – Fig. 3. The solid lines ("balance" and "relevance") are generated from Lucene, while the dashed lines ("refBal" and "refRel") are from Indri with query reformulation. We compare the results (of these four runs) with the median performance of the participants, together with the baseline provided by TREC.

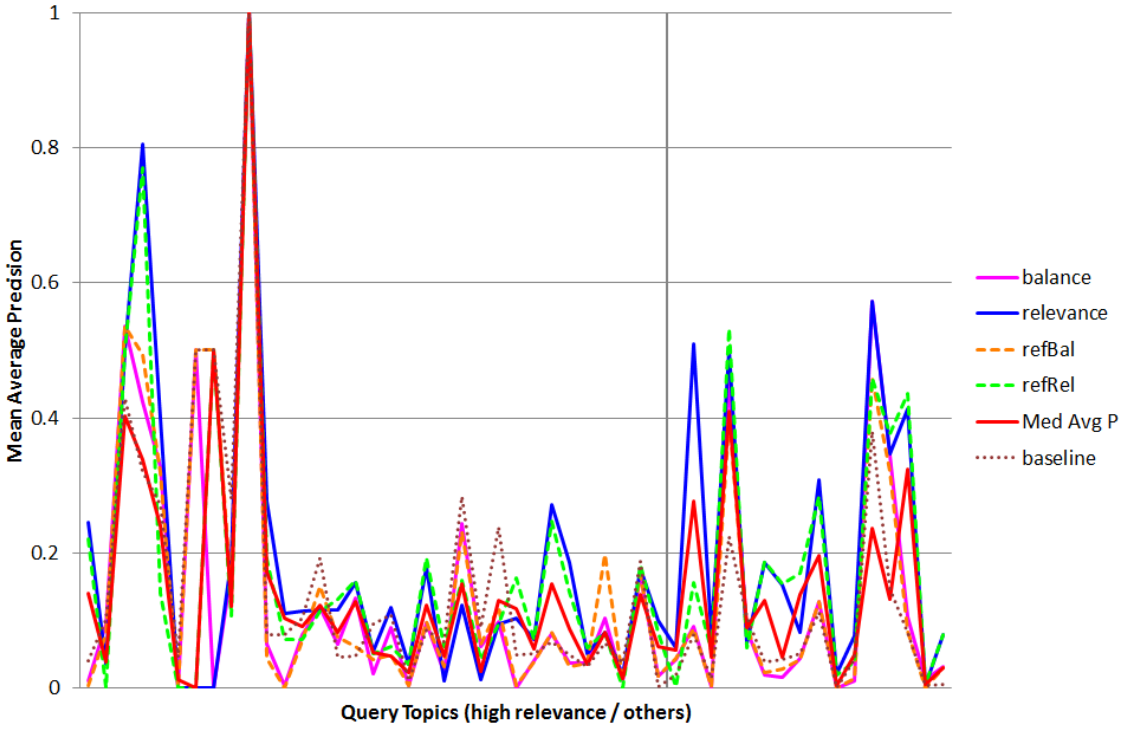

Fig. 1 Mean average precision on each query topic

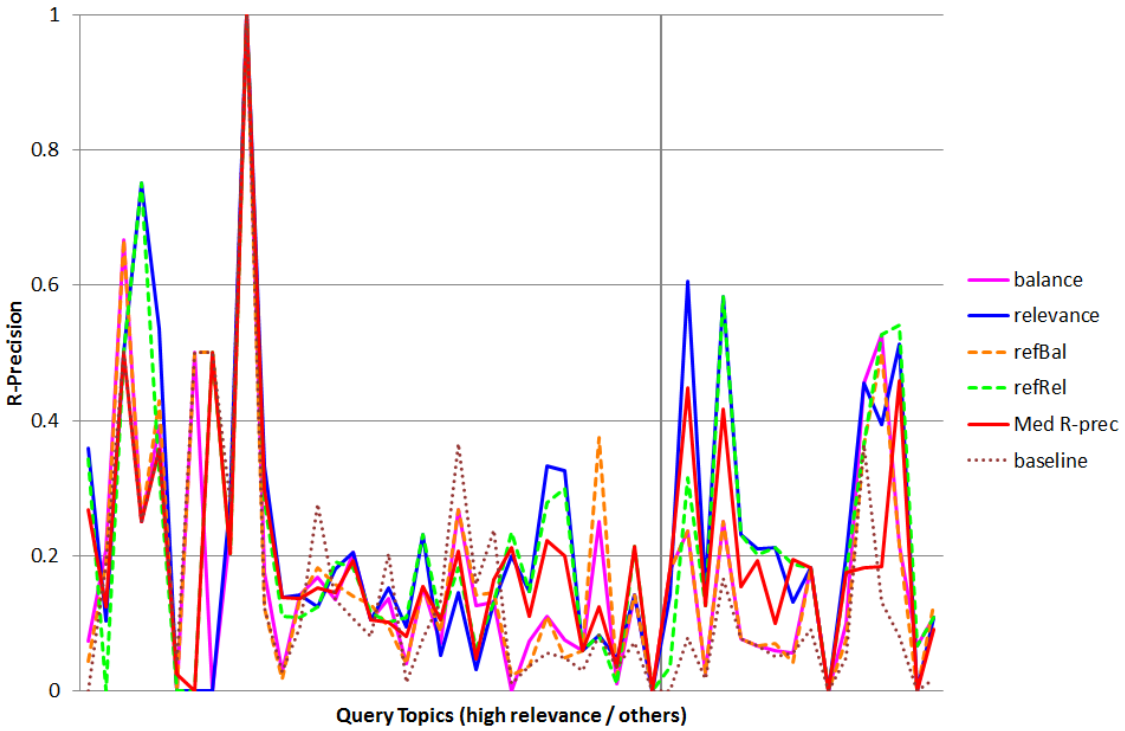

Fig. 2 R-Precision on each query topic

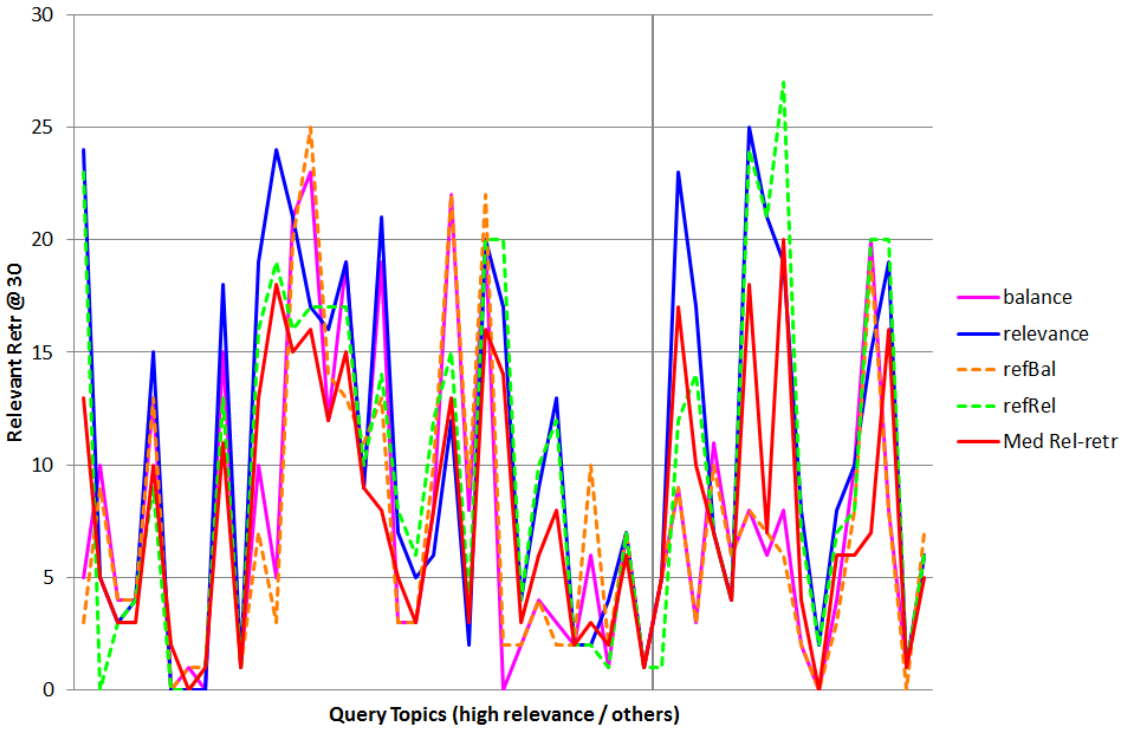

Fig. 3 Relevant Retr @ 30 on each query topic

From the figures we see clearly that our runs outperform the median performances in most topics.

Note that the baseline is based on Lucene without any further post-processing. Compared with the "relevance" line (Lucene + post-processing), we find that our post-processing methods improve the performance most of the time. However, in the cases baseline is better, it may due to the noises in the hashtags.

Another finding is that ranking with relevance is better than the balanced one. Although the task itself announced that the evaluation is from two aspects, we find the provided judgment (the official evaluation) considers the tweet IDs (represents the time a tweet is posted) retrieved in descending order as the rank order of the run. Under this consideration, our strategy may be designed differently to achieve a better performance.

For the comparison between with and without query reformulation, there is no significant difference in general.

6 Conclusions

In this paper, we describe our efforts in the participation of the TREC 2011 Microblog track. We design some specific methods both in preprocessing and post-processing, to fully utilize the feature of tweets, i.e. spam URLs and hashtags. Although the query model is simple, we find that these methods are helpful to discover relevant tweets.

For the future work, we expect to examine the provided judgment in detail to find out the features that bring a tweet to be relevant. Although tweet texts are short, there will be some specific characteristics that are helpful for tweets retrieval.

Acknowledgement

The NExT Search Centre is supported by the Singapore National Research Foundation & Interactive Digital Media R&D Program Office, MDA under research grant (WBS: R-252-300-001-490).

References

[1] Anqi Cui, Min Zhang, Yiqun Liu, Shaoping Ma. "Are the URLs Really Popular in Microblog Messages?". *Proceedings of the 2011 IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS2011)*. Beijing, 2011.

[2] Apache Lucene. http://lucene.apache.org/java/docs/. Visited on 1 Aug, 2011.

[3] Donald Metzler, W. Bruce Croft. "Combining the Language Model and Inference Network Approaches to Retrieval". *Information Processing and Management: Special Issue on Bayesian Networks and Information Retrieval*, 40(5): 735-750, 2004.

[4] Jimmy Lin. twitter-corpus-tools, https://github.com/lintool/twitter-corpus-tools. Visited on 1 Aug 2011.

[5] Anqi Cui, Min Zhang, Yiqun Liu, Shaoping Ma. "Emotion Tokens: Bridging the Gap Among Multilingual Twitter Sentiment Analysis". *Proceedings of the 7th Asia Information Retrieval Societies Conference (AIRS2011)*. Dubai, 2011.