ICTNET at Microblog Track TREC 2012

Bolong Zhu^{1,2}, Jinghua Gao^{1,2}, Xiao Han^{1,2}, Cunhui Shi^{1}, Shenghua Liu^{1}, Yue Liu^{1}, Xueqi Cheng^{1}
^{1}Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190
^{2}Graduate School of Chinese Academy of Sciences, Beijing, 100190

1. Introduction

There are two search tasksin TREC2012 Microblog Track[1], namely: Real-time Adhoc and Real-time Filtering.The Tweets2011 corpus is used again and last year's results can be used as first officiallylabeled data for any participants to traintheir models.In this year's track, the former task has60 new queriesgiven and the latter is first proposed.

In the Real-time Adhoc task, we use indri retrieval toolkit[2] to construct our retrieval system and propose a strategy of pseudo relevance feedback to expand original query,then we retrieved original tweets and their indri's scores as an important feature.Besides, we calculate lots of other features of these tweets, such abouturl, hash_tag, entropy, tfidf, bm25, language model[3] and proximity[4].At last, we use two learning-to-rank methods, specifically RankSVM[5] and ListNet[6], to combine all those featuresto sort them, returning the final ranked tweets to a specified query.

In the Real-time Filtering task, we assuming this task is similar with the topic tracking in Twitter Stream[7], we build up two filtering modelsbased on language modeland Vector Space Modelrespectively. Each model is initialized by the start query and its relevant tweets. For each new coming tweet, the model will decide whether it is under thetopic. If it is, we update the model to keep up with the development of the topic.

The rest of this paper is organized as follows. In Section 2, we discuss the preprocessing of Tweets2011 corpus. In Section 3, the main method to rank the search results in Real-time Adhoc task is discussed. In Real-time Filtering task, we describe two filtering models in Section 4. Experiment resultsare reported in Section 5. And in the last section, we draw conclusions about our work.

2. Data preprocessing

We use our last year's preprocessed corpus[8]. As tweets can be deleted or protected after posting, we additionally use an updating tweet ids list to remove tweets deleted from the corpus recently.Besides, word stemming is reprocessed and stop word removing is not processed as this will cause a large portion of information in such short text lost.Finally, there are 7,443,387tweets in our data set.

3. Real-time Adhoc

3.1 Query expansion

We leverage both internal and external expansions.In internal expansion, we assume the retrievedtop tweets which have themost ranking scores are more relevant to the topic.When viewing the top tweets as a document, we can estimate each word's new weight according to,

$$\mathrm{Weight(T)} = \frac{idf(T)}{Sum\_Score(D(K))} * \sum_{d \epsilon D(K)} score(d) * tf(d,T)$$

whereD(K) indicates the top-K returned tweets from the original query, Sum_Score(D(K)) is the sum of top-K tweets' scores, tf(d, T) is the term frequency of T in tweet d and idf(T) is the global inverse document frequency of term T in the corpus.

Besides, we use term concurrency frequency in the corpusas a second method to choose expansion words, their weights are estimated in following formula, where tc(T) indicates the frequency of term T concurrency with query terms, and it is normalized by$\sum_{t \in d} tc(t)$.

$$\mathrm{Weight(T)} = \frac{1}{Sum\_Score(D(K))} * \sum_{d \epsilon D(K)} score(d) * \frac{tc(T)}{\sum_{t \in d} tc(t)}$$

In external expansion, we use Google Search Engine to fetch result page of the original query. All wordsoccurred in top three links' titles are chosen to be the expansion words.

At last, we use Indri Retrieval Language to combine these expansion words and original query considering their new weights[9].

3.2 Features

We used two types of features, including static features (query independent) and dynamic features (query dependent, in both original and expanded queries, proximity features are only in original queries).For every result tweet, there are totally 51 features to be estimated (listing in the following tables). As the pages are limited, we can't describe every feature in details.Fortunately, lots of features can be described by their names, so we only choose some namelessfeatures to describe.

Static Features	Dynamic Features
has_url, has_hash_tag, has_rt;	covered_term_ratio;
url_slash_count_min,	tf_sum, tf_min, tf_max, tf_mean, tf_variance;
url_slash_count_max,	tfidf_sum, tfidf_min, tfidf_max;
url_slash_count_mean,	tfidf_mean, tfidf_variance;
url_len_min, url_len_max, url_len_mean;	boolean_model, vsm, bm25;
oov_ratio,unique_word_ratio,doc_len;	lmir_dir, lmir_jm, lmir_abs;
begin_with_at, indri_score, entropy;	min_proximity;
	mean_proximity;

In static features, indri_score is returnedfrom our Indri Retrieval System, oov_ratio is the ratio of out-of-vocabulary words count over the total words count, unique_word_ratio is similar to oov_ratio.

In dynamic features, tf value is query term frequencyin a tweet,tf-idf value is the query termtf-idf weight in a tweet. boolean_model, vsm, bm25, lmir (language model for IR) are the scores of the query to a tweet under these retrieval models.In lmir, three smoothing methods are used respectively, including DirichletPrior(dir), Jelinek-Mercer(jm) and Absolute Discounting(abs). We also estimate query proximity in a tweet.

3.3 Learning to Rank Methods

We utilize a simple linear learning-to-rank model to combine all these aforementionedfeatures. Given a query Q and a tweet D, a relevance score s(Q,D) is computed according to:

$$\mathrm{s(Q,D)} = \sum_{i}^{N} \lambda_i f_i(Q,D)$$

where N is the total number of features,$f_i(Q,D)$is a feature value and$\lambda_i$is a model parameter. Given a query Q, tweets D are ranked in descending order of their relevance score. To estimate model parameters, we use two types of learning-to-rank approaches, specifically RankSVM[5] for Pairwise approach and ListNet[6] for Listwise approach. Last year's queries and results are used for training, and 5-fold cross-validation is utilized.

4. Real-time Filtering

We assume this task is similar with the topic tracking in Twitter Stream[7], however, there are some differences in our approach.

A quality score is defined as a static weight for each tweet. The quality score is used to measure the probability of whether a tweet's content is meaningful, and is treated as a global weight for each tweet. We adopted the logistic regression model to calculate the quality scores. The features consist of several static features extracted inAdhoc task. And the tweets of the given 10 training topics are used as learning samples.

Two different retrieval models are utilized to implement two filters respectively. In the filter based on language model, we choose stupid backoff as the smoothing technique and "queue" as the history retention technique[7].In the filter based on vector space model, tf value is calculated in the "foreground" model and idf comes from the "background"model[7], and we also use the "queue" strategy to retain history information.

5. Experiments and Results

In Real-time Adhoc task, 60 queries are tested and four runs are submitted with different query expansions and different learning-to-rank methods.Tfidf query expansion is used in ICTWDSERUN1, and concurrency frequency query expansion is used in ICTWDSERUN2. In both ICTWDSERUN3 and ICTWDSERUN4, we use google search results as query expansion. RankSVMmethod is used in both ICTWDSERUN1 and ICTWDSERUN3, while LiseNetmethod is used in both ICTWDSERUN2 and ICTWDSERUN4.For each topic we return 10000 tweets in ICTWDSERUN1, and 1000 in both ICTWDSERUN2 and ICTWDSERUN3, but in ICTWDSERUN4 the returned tweets' number is estimated by a k-means method. The following table shows our results.

Table I. Experiments Results in Real-time Adhoc Task

RUN	group	P@30	MAP	manual?	RT?	docs?	extern?
ICTWDSERUN1	ICTNET	0.2384	0.2093	automatic	yes	no	no
ICTWDSERUN2	ICTNET	0.2339	0.1981	automatic	yes	no	no
ICTWDSERUN3	ICTNET	0.2113	0.1878	automatic	yes	no	yes

ICTWDSERUN4	ICTNET	0.2113	0.1650	automatic	yes	no	yes

In Real-time Filtering task, 40 topics are tested and four runs are submitted with different retrieval models and initialized strategies.VSM is used in both ICTNETFTRUN1 and ICTNETFTRUN2, and Language Model is used in both ICTNETFTRUN3 and ICTNETFTRUN4. To initialize the original topic we only use data in the corpus in ICTNETTFRUN1 and ICTNETTFRUN3, while we use the top-3 Google search results as an external resource in ICTNETTFRUN2 and ICTNETTFRUN4. The following table shows our results.

Table II. Experiments Results in Real-timeFiltering Task

RUN	group	Prec	Recl	F(0.5)	T11SU	manual?	RT?	docs?	extern?
ICTNETFTRUN1	ICTNET	0.1553	0.5020	0.1669	0.1265	automatic	yes	no	no
ICTNETFTRUN2	ICTNET	0.1513	0.5244	0.1630	0.1249	automatic	yes	no	yes
ICTNETFTRUN3	ICTNET	0.0000	0.3641	0.0000	0.0000	automatic	yes	no	no
ICTNETFTRUN4	ICTNET	0.0001	0.4933	0.0001	0.0000	automatic	yes	no	yes

6. Conclusion

In Real-time Adhoc task, we propose 51 features and use them in two learning-to-rank models which are trained by last year's results of the task.Finally, four runs are submitted with different query expansions. In Real-time Filtering task, we construct two filtering models to track topic in the stream of tweets, two runs are submitted with different tracking models.

Acknowledgements

We would like to thank all organizers and assessors of TREC and NIST. This work is sponsored by NSF of China Grants No. 60933005, No. 61100083 and No.61173064, and by 242 Program of China Grants No.2011F65.

References

[1] TREC 2012 Microblog Track http://sites.google.com/site/trecmicroblogtrack/

[2] Indri Retrieval Toolkit, http://www.lemurproject.org/indri/.

[3] J. M. Ponte and W. B. Croft.A language modeling approach to informationretrieval.SIGIR'93, Melbourne, Australia.

[4] Tao, T. and Zhai, C. An exploration of proximity measuresinformation retrieval. In the Proceedings of SIGIR'2007,pp.295-302, 2007.

[5] Joachims, T. Optimizing Search Engines Using Clickthrough Data,Proceedings of the ACM Conference on Knowledge Discovery andData Mining (KDD), ACM, 2002.

[6] Z. Cao, T. Qin, et al. Learning to Rank: From PairwisetoListwiseApproach, ICML 2007.

[7] J. Lin, R. Snow, and W. Morgan. Smoothingtechniques for adaptive online language models: Topictracking in tweet streams.In KDD, 2011.

[8] P. Cao, J. Gao, Y. Yu, S. Liu, Y. Liu, X. Cheng, ICTNET at Microblog Track TREC 2011, Proceeddings of the 20th Text REtrieval Conference (TREC 2011), 2011.

[9] Metzler D., Cai C., USC/ISI at TREC 2011: Microblog Track, Proceeddings of the 20th Text REtrieval Conference (TREC 2011), 2011.