ICTNET at Microblog Track in TREC 2014

Guoxin Cui[123], Fabin Shi[123], Xiaolei Liu[123], Xiaobo Hao[123], Xueke Xu[12], Yue Liu[12] , Xueqi Cheng[12]

1)Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190

2)Key Laboratory of Web Data Science and Technology,CAS,100190

3) University of Chinese Academy of Sciences , Beijing, 100190

{ cuiguoxin,shifabin,liuxiaolei,haoxiaobo,xuxueke }@software.ict.ac.cn; {liuyue,cxq}@ict.ac.cn

1 Introduction

Microblog track was first introduced in 2011 and we have participated in this task for 4 years [1,2,5]. This year's microblog track has two tasks. The first one, namely ad-hoc search task, is the same as usual. This task needs to retrieve all the tweets that are relevant to query Q before time T. Participants can access the corpus by official APIs. The second task is Tweet Timeline Generation(TTG) task. It is newly introduced this year and the main goal of it is to detect and remove the redundant tweets the first task retrieves.

This report is organized as follows. Section 2 mainly focuses on the data preparation. Section 3 is our methodology and framework of the ad-hoc search task. Section 4 focuses on the methodology of TTG task. Section 5 gives the final results of the two tasks.

2 Data Preparation

The twitter-tools[3] was downloaded from github. By using it we can interact with the service API to download the original tweets of each topic of each year in 2011, 2012, 2013 and 2014. We retrieved 10000 tweets for each query and stored them in separated file. We also stored the tweets of 2011, 2012 and 2013, since they were used as training data for our supervised framework.

We also used the TweetAnlyzer, the official supplied to do stemming and split the tweets into words. Since it has been shown that stop words removal might have a negative impact on the final ranking results, we didn't remove the stop words.

3 Ad-hoc Task Methodology

We define this task to be a re-rank problem. We have already downloaded the tweets of topics in 2011, 2012, 2013 and 2014. We use learning to rank model to do this re-rank problem. First we need to extract some features about one document and one query. We use the tool SVMrank [4] and the data of 2011 and 2012 which we have already known the relevance to train the model. Namely, we use data in 2011 and 2012 as train corpus and use data in 2013 as test corpus. At last we use all the data in 2011-2013 as train corpus to train the model and use this model to predict the data in 2014. Besides the features used in our previous work [5], we further consider features computed by the following methods.

3.1 Query Expansion

We use *Bo1* model[11] to get query expansion words. And we picked the *top-k* documents in one topic and use them to produce the expansion words. Every word *t* in the set has a weight w, and it is given by (1):$w(t) = tf * log\frac{1+p}{p} + log\,(1+p)$ (1)

where *tf* is the frequency of t in top-ranked documents and p is given by $\frac{F}{N}$ where F represent the term frequency of *t* in the whole corpus and *N* represents the total number of documents in the corpus. *Bo1* model can not only be used to give query expansion words for *BM25* score computing, but also can be incorporated into the language model.

3.2 Word Vector

We also use the word2vec [6] to get a feature value. We set the dimension of each vector of word to 200. For a query, the vector is computed by summing up the vector of each word's word vector weighted by the word's *tf-idf*. For the document, we do the same thing. Finally, we compute the cosine similarity of the two vectors and use this value as a feature.

3.3 Language Model

Besides the Bo1 model, we also used a mixture model to estimate query language model which regards a document as a mixture of theme [7]. It can be shown as:

$$p_d(w) = \lambda_B p(w|\theta_B) + (1-\lambda_B)\sum_{j=1}^{k}[\pi_{dj} p(w|\theta_j)]$$

$$log\, p(C|\Lambda) = \sum_{i=1}^{m}\sum_{d\in C_i}\sum_{w\in V}[c(w,d) \times log\, (\lambda_B p(w|\theta_B) + (1-\lambda_B)\sum_{j=1}^{k}\pi_{d,j} p(w|\theta_j)))]$$

We use uses Maximum Likelihood Estimation (MLE) with smoothing to learn tweet language model. The smoothing methods used are Jelinek-Mercer, Dirichlet, and Absolute Discounting. Finally, we use the *KL* divergence between the query and tweet language models to measure the relevance of the tweet to the topic.

3.4 BM25 model with term proximity

The most commonly used retrieval model is BM25. But BM25 model doesn't take care of the term proximity. So we proposed the minimum window BM25[8]. The main idea of the method is that if all terms of query appear in a small area, it's more likely to be relevant. There is another variant of BM25 named BM25PF [9] that we use. It also considers the term proximity. It combines the phrase frequency information with the basic bm25 model to rank the documents.

4 TTG Task Methodology

We apply two clustering methods on the Tweets return by ad-hoc retrieval system to capture the Timeline summary of certain relevant information. Single-Pass method and Affinity Propagation (AP) method are chosen for the sake of both performance and speed.

Single-Pass Clustering needs only one-time traversal of all the Tweets. Each new Tweet is compared with every formal Tweets in every cluster in Tweets similarity. If the largest similarity is larger than Similarity Threshold, the new Tweet is put in the cluster where the corresponding Tweet's is. If none of the similarity is larger than the threshold, the new Tweet is put in a new cluster which only contains itself now. The process ends until all the Tweets are put into clusters. The final clusters are the result of TTG Task.

AP Clustering is the state-of-art clustering method. This method maintain the Responsibility and Availability matrices which represent how well-suited Tweet A is regarded as Tweet B's exemplar and Tweet B is regarded as the follower of Tweet A. The matrices are carefully modified in each iteration until convergence. The final clusters are the result of TTG Task. Besides, this algorithm can also result in the exemplars of each cluster, which means the most representative Tweets of all clusters can be presented by this algorithm.

5 Experiments Results

For the first ad-hoc task, we submitted 4 runs. ICTNETRUN1 uses all the features mentioned above. ICTNETRUN2 doesn't use features generated by language model. ICTNETRUN3 doesn't use the features generated by word vector.ICTNETRUN4 doesn't use the feature of BM25PF score.

We set the parameter of SVM model to 0.3 and use the data in 2011, 2012 and 2013 as training set

to train the model. After that, we use the model to predict the rank of data in 2014. Finally, the result is shown in Table 1.

Table 1. Evaluation results for ICTNET submitted runs of task 1.

Run tag	R-Prec	MAP	P@30
ICTNETRUN1	0.4017	0.3534	0.5800
ICTNETRUN2	0.3734	0.3062	0.5109
ICTNETRUN3	0.4411	0.4139	0.6212
ICTNETRUN4	0.4369	0.4141	0.6242

For the second task, the result is shown in Table 2.

Table 2 Evaluation results for ICTNET submitted runs of task 2.

Run tag	unweighted_recall	weighted_recall	precision
ICTNETAP3	0.2234	0.4623	0.1792
ICTNETAP4	0.2528	0.4836	0.1702
ICTNETRUNSP3	0.2921	0.3959	0.1054
ICTNETRUNSP4	0.3410	0.4868	0.1029

6 Acknowledgements

We would like to thank all organizers and assessors of TREC and NIST. This work is sponsored by 973 Program of China Grants *No.2012CB316303&No.2014CB340401*, 863 program of China Grants *No. 2012AA011003&No.2014AA015204*,NSF of China Grants *No. 61232010&No.61173064* , and by the National Key Technology R&D Program Grants *No. 2012BAH39B04&No.2012BAH46B04.*

Reference

[1] Peng Cao, Jinhua Gao, Yubao Yu, Shenghua Liu, Yue Liu and Xueqi Cheng. ICTNET at Microblog Track TREC 2011. In Trec 2011.

[2] Bolong Zhu, Jinhua Gao, Xiao Han, Cunhui Shi, Shenghua Liu, Yue Liu and Xueqi Cheng. ICTNET at Microblog Track TREC 2012 In TREC, 2012.

[3] Jimmy Lin. Twitter Tools. https://github.com/lintool/twitter-tools, 2012

[4] Thorsten Joachims. Training Linear SVMs in Linear Time. In Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2006.

[5] Jinhua Gao, Guoxin Cui, Shenghua Liu, Yue Liu and Xueqi Cheng. ICTNET at Microblog Track TREC 2013 In TREC, 2013.

[6] www.deeplearning4j.org/word2vec.html

[7]Zhai C X, Velivelli A, Yu B.A cross-collection mixture model for comparative text mining[C]//Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2004: 743-748.

[8]F. Guan, X. Yu, Z. Peng, H. Xu, Y. Liu and L. Song. ICTNET at Web Track 2009 Ad-hoc Task. In Proceedings of theEighteenth Text REtrieval Conference, Gaithersburg, Maryland, November 17-20, 2009

[9]Yadong Zhu, YuanhaiXue, JiafengGuo, YanyanLan, and Xueqi Cheng, Xiaoming Yu. Exploring and Exploiting Proximity Statistic for Information Retrieval Model. Proceedings of the 8th Asia Information Retrieval Societies Conference (AIRS12), pp: 1-13, Tianjin, China, 2012.

[10] Yasuhiro Fujiwara, Go Irie, Tomoe Kitahara. Fast Algorithm for Affinity Propagation[C]. Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, 2010. 2238-2243.

[11] Giambattista Amati. Probabilistic Models for Information Retrieval Based on Divergence from Randomness. PhD thesis, Department of Computing Science, University of Glasgow, 2003.