ICTNET at Temporal Summarization Track TREC 2014

Lei Chen[123], Hainan Zhang[123], Siying Li[123], Zhiyuan Ji[123], Qian Liu[123], Yue Liu[12], Dayong Wu[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

{chenlei2013, zhanghainan, lisiying, jizhiyuan,liuqian}@software.ict.ac.cn; {liuyue,wudayong,cxq}@ict.ac.cn

1 Introduction

Temporal Summarization task is a standard Information Retrieval problem. The goal of the task is to generate sequential update summarization, which are useful, new and timely sentence-length updates about a developing event[1]. The event refers to a temporally acute topic, and each topic contains the start time and end time. There are more event types than the last year. They are accident, bombing, hostage, impact event, protest, riot, shooting and storm. Formally, given the time-ordered corpus, the query and the relevant time range, our system outputs a list of relevant sentence identifiers.

2 Our Approach

2.1 Data Preprocessing

The first step of preprocessing is downloading the filtered data from the TREC-TS website[2]. Then we use the GPG key, which can be obtained after submitting the TREC agreement, to decompress the data. In order to deal with the big scale of the data downloading from the website, we filtered the data in two steps. First, any documents that were published out-with the time periods of the 15 events from the TREC-TS 2014 track topics were removed, i.e. only documents with timestamps between the start and end tag for one or more TREC-TS 2014 topics were kept. Second, we build index for the documents whose titles and contents were likely to contain at least one word of the query. Then we use the useful open source indexing tool Lucene to build the index of the filtered data to further reduce the amount of data.

2.2 query analysis

We also analyze the 15 test queries. There are 6 test query belonging to protest category, 3 belonging to storm, and each have 1 test query of category accident, bombing, riot, hostage, impact even, shooting. Timeframe of two query with same category may have crossed. Such as query22 and query23, belonging to protest category, and time overlap. So, we cannot judge whether it is related with query just by topic category. It make getting related document sentences matching the query more difficulty.

2.3 Idea

We consider not only the accuracy but also recall. Matching to the query can get more accurate of the search result, but the recall is extremely low. In order to improve the recall, query expansion is almost an essential process. Thus we expand the test query, and then use the expended query on the matching method. Using query expansion method, recall has been greatly improved. For the expanded query is still relatively strict, not much loss of accuracy.

But It have two drawbacks . They are as followed:

1. Limited by the degree of expansion, the recall amount of relative result is still low

2. It may miss some related documents and recall some not related documents using method of matching to the query.

Now we talk about the second. Missing related documents is because the document does not appear the query or expansion query. So it will miss the document though it is relate to the query. Recalling not related documents because the page of the document is not a regular document. There may be a lot of advertising, or other related content recommendation. If the query appeared in advertising or related recommendation, it may cause fault result be recalled and reducing the accuracy. So, I consider to use the topic of the document to judge if the document is related to the test query, not just using matching expanded query words. We use some keyword as feature to describe topic. To select keyword feature, we use LDA[3] and SVM[4] model.

2.4 Generate Sequential Update Summarization

After data preprocessing, we use two methods to get the final results. They are LDA and SVM.

2.4.1 Latent Dirichlet Allocation Module

LDA is an unsupervised learning method. We use the LDA to find the latent semantic topic. For each query, we used each document which in the time range as the input of the LDA model, and we would get ten different topics. Then we build the all-query index as the standard documents which title have all of the query words. For each topic we score the all-query index and get its scores. Comparing the scores of the ten topics and the all-query score, we can identify the most similar topic of the query in the ten topics from the result of LDA automatically.

The LDA model can generate a list of keywords, in which every word has its own weight. Thus we can use the list of keywords to score the sentence and drop the sentences with score lower than the threshold. The keyword scoring method is described as following:

$$Score(S_i^k) = \frac{1}{|S_i^k|}\sum_j value(w_j), w_j \in S_i^k,$$

where $\text{value}(w_j)$ is the score of word w_j in the list of keywords, S_i^k is the i^{th} sentence of topic k. If the score of sentence is larger than the threshold, we put this sentence into the result of our method. Otherwise, drop it.

After the process of sentences scoring, we need post-processing on the results of our method. On the one hand, we have to sort the results timely due to the importance of expected gain with latency-discounted. We use a simple distance function to find the same sentences and drop the latter one with time stamp. On the other hand, we drop the sentence with the length shorter than 3 and longer than 40. Because the short sentences and the long sentences may not contain the important information.

2.4.2 Support Vector Machine method

SVM is a supervised learning classifier. Its classification results often depend on the features chosen for SVM. According to this idea, we marked some corpus for training SVM classifier for each query. We used chi-square to select keywords for each query as SVM features. As the same time, we can also get the weight of the keyword. Then, we train classifiers for each test query. After getting the classifier, we classify documents within the time range of test query, judging whether the document is related to the query. If related, remain it for further processing. Or discarded directly. After classifying the documents, we get the related documents. Then we calculate the sentence score of the related document as the method of LDA use.

3 Results

We submitted a total of four runs. Run1 is the most basic result of LDA module. Run4 is the improved version of run1. Run4 removes the sentences that do not contain any of the top 100 key words. Run2 and run3 are the results of SVM module. We compare these runs as Table 1.

Table 1. The comparison of our four runs.

Run ID		Latency Gain	Latency Comp.	HM
run1	AVG	0.0082	0.4929	0.016
	STD	0.0063	0.2201	0.0121
	MIN	0.0015	0.1537	0.0029
	MAX	0.0219	0.965	0.0422
run2	AVG	0.0458	0.0773	0.0311
	STD	0.1152	0.101	0.0473
	MIN	0	0	0
	MAX	0.468	0.339	0.1543
run3	AVG	0.0632	0.105	0.053
	STD	0.1265	0.1366	0.0882
	MIN	0	0	0
	MAX	0.468	0.4515	0.3327
run4	AVG	0.0092	0.4927	0.0178
	STD	0.0069	0.2201	0.0131
	MIN	0.0023	0.1537	0.0046
	MAX	0.0238	0.965	0.0456

4 Conclusion

This paper describes the two methods and the complete process of us when doing this temporal sequential update summarization task. We use both supervised and unsupervised method to accomplish the requirements of the task of this year.

5 Acknowledge

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.2013CB329602*, 863 program of China Grants *No. 2012AA011003&No. 2013AA01A213*,NSF of China Grants *No.61232010&No.61173064*, and by 242 Program of China Grants *No. 2013F099*, and by the National Key Technology R&D Program Grants *No.2012BAH39B02&No.2012BAH39B04*.

Reference

[1] Aslam, Diaz, Ekstrand-Abueg, McCreadie, Pavlu, Sakai.Temporal Summarization. Available: https://38309103-a-62cb3a1a-s-sites.googlegroups.com/site/temporalsummarization/trec2014-ts-guidelines.pdf.

[2] TREC-TS-2014F dataset. Available: http://s3.amazonaws.com/aws-publicdatasets/trec/ts/index.html.

[3] Blei D M, Ng A Y, Jordan M I. Latent dirichlet allocation. The Journal of machine Learning research, 2003, 3: 993-1022.

[4] Chang C C, Lin C J. LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology (TIST), 2011, 2(3): 27.