THUIR in TREC2003: HARD Experiments

[1]Liang Ma, [1]Wei Tan, [1]Qunxiu Chen, [1]Shaoping Ma
[2]Shuicai Shi, [2]Shibin Xiao, [2]Hongwei Wang, [2]Hongjun Wang
[1]State Key Lab of Intelligent Tech. & Sys., CS&T Dept, Tsinghua University, Beijing 100084,China
[2]TRS Open Software Laboratory, Beijing Information Technology Institute, Beijing, China

maliang00@mails.tsinghua.edu.cn

Abstract

In this paper, we describe ideas and related experiments of Tsinghua University IR group in TREC-12 HARD Track. In this track, we focus on an automatic delivering mechanism, which combine the existing IR methods and can provide a quick retrieval solution for the practical environment. The final official evaluation show the old ways perform not well, but we think the experiment data will be helpful in evaluating the new ideas developed by other teams.

1. Brief Introduction

As a new evaluation track, HARD is designed to find an effective way to locate the search focus precisely from the data coming from the user, including his/her additional information (such as the he/his background) and an interactive input, so as to provide better retrieval result to the original query request.

In this track, the key issue is to find the real search focus. There can be two ways to finish it: manually and automatic way. Though the former usually can provide a satisfied performance, we think the automatic way is more useful in practical use and thus try to devote us in this way.

In following sections, we introduce what we did in our research work and give the final evaluation result. Some further research work done after final TREC submit is also listed.

2. Construct Baseline Run

We get our baseline run (only with document) using the initial query by a BM25 TF*IDF scoring schema. It is a popular method that is fast and practical. The special treatment is only used for initial query: For each topic, the query is constructed simply by the task description (The detail restriction for none-relevant document are ignored). For the search items, different weights are set according to their location(such as description field) and importance in the task description. Also, there is no positive training documents are used to refine the query, because usually the training resource is unlikely to be provided for various immediate search requirements in Web IR.

3. Focus Probe

In focus probing, we try to find the search focus of the user input. In this period, there are two

missions we did:

3.1 Finding potential search items

In our Clarification Form, all the potential search issues to be confirmed by user are listed with checkbox, together with a text field to fill if he/she finds there are something we missed. These search terms are presented as some keywords or phrases instead of long statements extracted from web pages. Some existing technologies, such as complex passage analysis or do self-learning from related training resources, seems to be good ideas but time-consuming. Here we choose a fast mechanism to extract them automatically. They are got from two ways follow:

(1) The kernel words/phrases in topic description. We parse the description and get presentive words/phrase set from each fields, then all the set are combined and those words/phrase existing in multi set are thought as kernel words/phrases.

(2) Terms with high statistical weight in top-100 ranked documents in search result. But only the terms in the title and the first paragraph(not the whole passage) are calculated, for there should be more focus-related words in these two sections. To keep the search deviation under control, we limit the potential search terms up to 10 issues.

Compared to other methods, our idea is efficient in finding the potential search terms, also it doesn't require any training resource, therefore it is feasible when applied in a practical use, but the accuracy of this method has not been proved to be very satisfied.

3.2 Locate the Desired Focus

We locate the desired focus from:

(1) **Returned Clarification Form from LDC.** Since the returned Clarification Form has been processed by search user, all the words/phrases in selected checkbox and the content filled in the additional text field are thought as desired search focus.

(2) **The *searchitem* filed in metadata.** Only this field in metadata is used to provide short search terms, other fields are all ignored.

4. Refine the Query

Based on the initial queries used in the baseline run, we improve them using the desired focus newly located. But different update styles are used according to how the focuses are located.

(1) Focus from returned Clarification Form

The words/phrases in each selected checkbox and the filled content in return CF are thought as one search focus. Based on the kernel terms in initial query and the current search item, a sub-query is constructed for a specific search focus. Then the initial query is divided into several queries for different search focus.

(2) Focus from *searchitem* field

All the search terms in the **searchitem** field are simply added to the initial query as new weighted terms. They are merged using Rocchio-like feedback mechanism.

From the above improvement, we construct the search query for the final run.

5. Refine the Query

5.1 Return type detection

For various topics, the user want to receive different search result: document, passage and sentence. We decide the return type by following rules:

- We return document if topic require so.
- For passage and sentence, we usually return the result based on paragraph. For passage, usually there are one or two paragraphs are included. For sentence, it is nearly impossible to present an efficient result in such rough retrieval, therefore a paragraph will be more meaningful.
- If any type is welcomed, we analyze the topic description and decide the result should be passage or document. For example, if there are some words ' the document should….'in the description, then we think a document should be returned.

5.2 Paragraph level indexing

For test corpus, we built an index based on the document level. Since the paragraph will also be returned, we create a new index based on the paragraph. Each paragraph in the document is taken as a single passage and indexed. For some short paragraphs, we merge them to the neighbor paragraphs until the length of this paragraph to be indexed is large than average paragraph length of the corpus.

5.3 Result merging for final submission

All the improved queries are submitted in the document index or paragraph index according to their return type. For topics that return passage and sentence, we also do the retrieval work in the document index. Before getting the final result, we do the following work to the scored list:

- Merge by sub-query: For the topics which have sub-queries presenting different search focus, the final retrieval result is the combination of all of sub-queries, and the scored item is ranked as their order in baseline run(for passage and sentence, they use the order of their host document).
- Document detection for passage and sentence: we return paragraph when topic require a passage or sentence. To keep out of the noise paragraphs, for a retrieved relevant paragraph, only its host document is also ranked as the topic-relevant that can we set it to the returned final result.

6. Final Submission and Evaluation

We finally submitted three runs which expand query using different data source (but with same weighting/scoring parameters for query). For each run, the detail parameters and its evaluation result(also include the baseline run) are listed in table 1 and table 2.

Table 1. Parameters Setup in Each Run

Run	Fields used in task description	Use Clarification Form	Use Metadata	Merge Result
Baseline Run	Title Description Narrative			
TUCSHARD1		Yes	Yes	Yes
TUCSHARD2		Yes	No	Yes
TUCSHARD3		Yes	Yes	No

Table 2. The TREC Evaluation Result for Each Run

Run	Evaluation			
	Passage level		Document level	
	R-Precision	F-score at 100 passage	R-Precison	
			Hard-rel	Soft-rel
Baseline Run	0.1235	0.1294	0.1960	0.2560
TUCSHARD1	0.1868	0.1396	0.2148	0.2818
TUCSHARD2	0.1655	0.1296	0.2012	0.2627
TUCSHARD3	0.1868	0.1396	0.2138	0.2711

7. Conclusions

The evaluation result tell us the clarification form, in lifting the query precision, work better than the metadata. Some of our work later on constructing clarification form using certain cluster algorithm provided us more satisfied result. Also we noticed the result merge seems an effective tool, especially in small amount of documents returned.

Reference

1. Robertson, S. E., and Walker, S. Okapi/Keenbow at TREC-8. In Proceedings of the *TREC-8*. Maryland. 1999.
2. S.E. Robertson, H. Zaragoza, M. Taylor, Microsoft Research Ltd. HARD Track Overview in TREC 2003 High Accuracy Retrieval from Documents. In Notebook of TREC-2003.
3. T. Morton. Using Coreference in Question Answering. In Proceedings of the TREC-8. Maryland. 1999.
4. Y. Ogawa, H. Mano, M. Narita, S. Honma. Structuring and Expanding Queries in the Probabilistic Model. In Proceedings of the TREC-9. Maryland. 2000.
5. R. B. Allen, P. Obry and M. Littman. An interface for navigating clustered document sets returned by queries. In Proceedings of the ACM Conference on Organizational Computing Systems, 1993