York University at TREC 2007: Enterprise Document Search

Yu Fan[1] *Xiangji Huang*[2]
iamfanyu@gmail.com jhuang@yorku.ca

[1] *Department of Computer Science & Engineering*
York University, Toronto, ON, Canada

[2] *School of Information Technology*
York University, Toronto, ON, Canada

Abstract

York University evaluated a prepcessing approach for this year's enterprise document search task. With different parsing tools, we create two data sets. Based on each data set, we generate two official runs. Their results demonstrate that the removal of raw data in preprocessing stage has a negative impact on the retrieval performance.

1 Introduction

This year we use the Okapi system for our enterprise document search experiments. In contrary to last year's data post-processing approach, our focus in this year's enterprise track was on data preprocessing.

One of the main challenges in enterprise search is to deal with a large number of data type variation. Considering the enterprise search as a chain of functional components with data flow through, an obvious bottle neck of all-time is preprocessing. One of our major objectives is to develop a methodology to preprocess the data.

This year, the enterprise track use a new data collection. The task is to find missing page or a set of pages contains keywords that can help to create an overview page about a given topic.

We assume that an overview page must be very "human-oriented" in comparison to machine generated pages. "Human-oriented" files intend to be "natural expression of opinons". For example, emails or lecture notes are much more human-oriented than a SQL report page or a database table file or dynamically generated Web forms. This assumption is not extremely logically sound. However, it does give us a start point to implement a preprocessing strategy

First, we identify relevent files that are also human-oriented. After examining the data collection, we find that there are many documents with an empty body and a source path that links to an application file such as MS Word file, or a presentation slide, or a pdf file. They were crawled but not included in the data collection before we trace their location and

download their contents. This reminds us the challenge of dealing with data type variation and the importance of pre-processing methodology.

Also, we find there are a lot of documents that are database tables. Since we focus on finding human-oriented files, these files are in general not of interst to us. Therefore, these database tables can just be placed into our to-be removed list.

2 Our System and Method

2.1 Okapi

We used Okapi BSS (Basic Search System) as our main search system. The weighting function used is BM25 [2]. For a given term t, a query q and a document d within a collection of documents, the weight w of d with respect to q and t is calculated by following formula:

$$w = \frac{(k_1+1)\times tf}{K+tf} \times log\frac{N-n+0.5}{n+0.5} \times \frac{(k_3+1)\times qtf}{k_3+qtf} \quad \oplus \quad k_2 \times nq \times \frac{(advl-dl)}{(advl+dl)}$$

Table 2.1 BM25 variables references

Variable	description
N	total number of indexed documents in the collection.
n	number of documents containing the term t. ($n \leq N$)
tf	within document term frequency of t in d.
qtf	within query term frequency of t in q.
nq	number of query term (query length).
dl	length of document d.
avdl	average length of all indexed documents in the collection.
K	$k_1 \times (1-b+b\times \frac{dl}{avdl})$
K1,k2,k3,b	constants, used to tune the system.

BM25 weighting scheme is the core of Okapi system. Relevance judgment of terms and documents are made based on calculated weights shown above.

2.2 Data Preprocesing and Query Expansion

As discuss above, we examine a fraction of the data, for each identified data type, either re-collect the doc's body or mark the doc as to-be removed. We also investigate the effect of different parsers on the raw data. Finally, we generate two data set. One contains 370,715 documents. The other one contains 360,715 documents. These two data sets are used to create our four official runs.

Raw data are converted into its original form: one document per file. A well-written parser will generate an 'exch' file by a single run on the raw data. This is not true for the entperise track due to the fact that enterprise data type is never homogeneous. For query expansion, we add a narritive term list to query term list and use the average weight of query terms as a threshold. Query expansion, in gereral, does make a positive contribution to the retrieval performance.

3 Experimental Results and Discussions

We used a java-based parser to process the raw data and create the first dataset. This parser removed all the XML tags and HTML tags. A text-base browser combined with perl script created the second dataset. Database files were removed from the second dataset. Then on each dataset, we generated two runs. One run was generated using query from the topic file. The other run was generated using queries expanded with narritive terms from topic files. We would like to investigate how the data post-processing (such as query expansion) is influenced by the data pre-processing.

The official runs were generated using Okapi 2.41 and the parameters for BM25 were set to 3, 0, 8 and 0.55 for $k1$, $k2$, $k3$ and b respectively. Table 3.1 shows a comparison of parser during preprocessing. Table 3.2 shows the effect of query expansion.

Table 3.1 Title only query, Port's stemming with gsl.lemurl

Run	MAP	R-prec	Description (Table 3.1)
york07ed1	0.4405	0.4484	a Java parser,blindly remove tags query only 370715 docs,keep 10000db files
york07ed2	0.4270	0.4480	lynx dumping tool, with perl script remove page overhead,query only,360715 docs

Table 3.2 Query expanded with narrative, Port's stemming, gsl.lemurl

Run	MAP	R-prec	Description (Table 3.2)
york07ed1	0.4405	0.4484	Query only, no expansion. 370,715 docs
york07ed4	0.4536	0.4553	Same as york07ed1, with narritive query expansion
york07ed2	0.4270	0.4480	query only, 360,715 doc with db file removed,no expansion
york07ed3	0.4237	0.4351	Same as york07ed2, with narritive expansion

4 Conclusion and Future Work

As the results show, more documents always give better performance although it seems to us that these removed documents have nothing to do with overview papers.

Also, recovering of those so-called human-oriented files did not give us obvious performance gain. Especially, when the number of recovered documents is not significant comparing to the size of the dataset.

Acknowledgement

This research is supported in part by the research grant from the Natural Sciences and Engineering Research Council (NSERC) of Canada.

References

1 S. E. Robertson, J. K. Sparck (1976). *Relevance Weighting of Search Terms. Journal of the American Society for Information Science 27, May-June 1976, p129-146*

2 M. Beaulieu, M. Gatford, X. Huang, S. E. Robertson, S. Walker and P. Williams (1996). *Okapi at TREC-5. Proceedings of 5th Text REtrieval Conference*, pp. 143-166, 1996.