IR Programming assignment 1 Report

B04703001 蔡明宏

VSM

(1) TF-IDF model

$$\sum_{t \in Q,D} \ln \frac{N - df + 0.5}{df + 0.5} \cdot \frac{(k+1)tf}{k(1 - b + b\frac{dl}{avdl}) + tf)} \cdot \frac{(ka+1)qtf}{ka + qtf}$$

I use the above equation to evaluate the score of a document given a query Q. I can use “file list” to get N, “inverted-file” to get document frequency(df) and term frequency(tf). The remaining work is to define term frequency of a query and decide the hyper parameters k, b, and ka. I would introduce how I define qtf in the next subsection. I would discuss the experiment of the hyper parameters in the Results of Experiments section.

(2) How to define term frequency of the query

In the query file, it has extracted the concepts of the query for us. I choose to precess concepts to get the term frequency of the query, because they provide much more precise information comparing to raw question and the narrative. I divide all the concepts into uni-gram and bi-gram, because in Chinese language, lots of words consist of single character or two characters, so in general, the approach based on gram units can capture the semantics precisely. Fortunately, the inverted-file also indexes the documents by uni-gram and bi-gram.

Relevance feedback

(1) How to define the relevant document?

I first calculate scores among all the documents. Then I pick top FEEDBACK_DOC_NUM documents to be my relevant documents. One thing I want to mention here is that the value of FEEDBACK_DOC_NUM would affect performance because it would increase the number of the processing terms dramatically.

I only use relevant documents to feedback the retrieved documents. Because correct documents are more important than irrelevant documents.

(2) How I update the terms vector of query using relevant documents.

After I pick up the top FEEDBACK_DOC_NUM documents, I calculate the centroids of these documents. Then I pick top FEEDBACK_TERM_NUM term id based on the weight of the centroid. I do not consider all the terms, because it might cause performance issues and the biased retrieved results. Then I use the below equation to update the term vector of the query.

$$\overrightarrow{q_m} = \alpha \overrightarrow{q} + \frac{\beta}{|D_r|} \sum_{\forall \overrightarrow{d_j} \in D_r} \overrightarrow{d_j}$$

Results of Experiments

(A) VSM hyper parameters: Okapi_k, Okapi_b, and Okapi_ka

(1) Base line - Random chosen value

Okapi_k	Okapi_b	Okapi_ka	MAP
1.5	0.75	500	0.813229

(2) Increase normalization of the document term frequency

In my intuition, increase penalty of the document term frequency can improve performance, because numbers of terms contained in a document can be large and term frequency can be less significant for the similarity when the value is high.

Okapi_k	Okapi_b	Okapi_ka	MAP
1.75	0.75	500	**0.814023**
1.9	0.75	500	**0.814409**
2.0	0.75	500	**0.813875**

(3) Decrease normalization to the query term frequency

Query is the relatively small document. The frequency in a small document has great significant impact on the semantic when value is small

Okapi_k	Okapi_b	Okapi_ka	MAP
1.9	0.75	**250**	**0.814442**
1.9	0.75	**125**	**0.81443**
1.9	0.75	**175**	**0.814466**

(B) Relevance feedback hyper parameters: DOC_NUM, TERM_NUM, Alpha

(1) Base line - random chosen value

FEEDBACK_DOC_NUM	FEEDBACK_TERM_NUM	Alpha	MAP
20	100	0.8	0.810716

(2) Top DOC_NUM document

I found out that my first 12 retrieved documents has high precision in general, so I only choose top 12 documents to be the feedback target.

FEEDBACK_DOC_NUM	FEEDBACK_TERM_NUM	Alpha	MAP
12	100	0.8	**0.818807**

(3) Alpha

I observe that updated vector improve the quality of the retrieved documents. I try to add the weight of updated vector part.

FEEDBACK_DOC_NUM	FEEDBACK_TERM_NUM	Alpha	MAP
12	100	**0.75**	**0.819462**
12	100	**0.7**	**0.816621**

(4) Top TERM_NUM Term

The average term frequency of query is about 70. Because I think the concepts have already covered the important semantic, I do not want to add too much term in the query. I want to focus on weight feedback on the original term.

However, after experiment, it seems that 100 is already a perfect value.

FEEDBACK_DOC_NUM	FEEDBACK_TERM_NUM	Alpha	MAP
12	**90**	0.75	**0.816825**
12	**110**	0.75	**0.819146**

Discussion

Lazy calculation

If the value is not needed right now, lazy evaluation can improve the performance. For example, the term vector of document. When evaluating the score of the document, only the terms which also contained in the query term vector are needed, which is the small portion in a document. When I do relevance feedback, only the top FEEDBACK_DOC_NUM documents need to computer their complete term vector. Compared with computing complete term vector in all documents, the program can perform faster in this approach.