Web Retrieval and Mining Assignment #3

B0591025 Yu-Neng Wang

VSM Model

1. **Vocabulary and Inverted-file**
   I use the given vocabulary and inverted-file, and do the following to process query and document.

2. **Document**
   I walk through the inverted file and keep a dictionary which stores the terms and their frequency for every document. These dictionaries can be view as the "vector" of documents.

3. **Query**
   Remove punctuation mark and extract unigram and bigram from the query, and keep a dictionary similar to dictionaries in documents to store the terms and their frequency. Besides, for terms extracted from section "concepts", I would amplify their frequency because they are keywords in the query.

4. **Weighting Scheme**
   I use the popular Okapi/BM25 weighting, and set k1=1.5, b=0.75, k3=100 as default parameters. Details of choosing parameters will be discussed in section "Experimental Result".

Rocchio Relevance Feedback

I only use top k documents as relevant documents and don't consider non-relevant documents. The feedback method is merging the dictionaries (vectors) of relevant documents into the query's dictionary (vector). That is, if a term is in both a relevant document and the query, its new term frequency will equal sum of its term frequency in the document and the query; if a term is not in the query but in a document, the term will be inserted into the query's term dictionary and has term frequency equal its frequency in the document.
For k, I choose k=10 for experimental results. k being too large or too small will cause degeneration on solution qualities. I've also tried to do feedback more than once, but didn't improve the qualities either.

Experimental Result

The experiments are conducted on the following parameters. Query processing: concepts_weight=5; Okapi/BM25: k1=1.5, b=.075, k3=50; Rocchio Feedback: feedback=true, feedback_size=10, alpha=0.8, beta=0.2; and perturb one of them.

concepts weight	1	5	10
train MAP	0.811	0.803	0.799
public test MAP	0.799	0.815	0.805
private test MAP	0.753	0.761	0.764

Table 1: Perturb concepts weight

k3	2	50	100
train MAP	0.789	0.803	0.804
public test MAP	0.792	0.815	0.815
private test MAP	0.728	0.761	0.761

Table 2: Perturb k3

feedback size	0	5	10	15
train MAP	0.811	0.809	0.803	0.811
public test MAP	0.780	0.805	0.815	0.795
private test MAP	0.749	0.752	0.761	0.747

Table 3: Perturb feedback size

alpha	0.2	0.5	0.8
train MAP	0.773	0.797	0.803
public test MAP	0.800	0.813	0.815
private test MAP	0.689	0.750	0.761

Table 4: Perturb alpha & beta (beta=1-alpha)

Discussion

Table 1 shows that increases concepts weight increases MAP. This is consistent with my assumption that "concepts" are keywords thus help retrieval process. Table 2 shows that k3 should not be too small. Large k3 emphasizes the importance of term frequency in query and can help enhance retrieval precision.

Table 3, 4 show the impact of parameters in Rocchio feedback. We can observe that feedback indeed help to retrieve documents. However, too much feedback documents might be misleading and result in retrieval quality degeneration. Besides, Table 4 shows that large alpha results in better retrieval quality. It indicates that although feedback can help retrieval, the retriever should still follow the original query as main guidance and feedback documents as reference.

Another interesting thing to discuss is the run time. I wrote python code to build the retrieval model and beat the strong baseline. This part is not very hard. Nonetheless, how to compress the run time so that it can finish within 5 minutes cost me a lot of time. I was not sure that it was because of my poor coding ability or python's low efficiency resulted in the lengthy run time. In the end, I used the Cython package to compile my python code and ran within 5 minutes.