IR program 1 report R02922083 邵元輔

VSModel and file setup

The VSM I use is consist of 2 parts : 1st I use inFilehendle.py to convert in-file to a 2-dimension matrix with column in term space and row in file space. The term space is consist of the vocabulary 2-gram described in vocab model, also the 1-gram is considered in terms therefore there are about a million terms to be considered. The TFIDF model I used is BM25. I calculated the doc part of BM25 and save the matrix for further use. In addition, docLenEXtraction.py is used to preprocess the doc length for TFIDF model. 2nd part I use queryHandle.py to calculate the score and MAP for parameter tuning. In this python file, I convert each query into a term vector of BM25 query part and separate each description in a list. And combine 4 term vectors in a complex combination with parameter tuning. For example, a title might multiply concept then add question to get the query vector. After that, multiply the file matrix and query vector to get the score list of each query. Note that the matrix and vector is very sparse and is handled with scipy.sparse.csr_matrix class to save the memory space and calculation time. Since there are some ans-trian doc fiel not in the file list I set the notfound file to be founded after top100 files.

The Feedback system is set with option=true, and I use the ans-train to slightly update the input query to a new query close to the correct query by adding the relevant doc term vector on the original query. Note that the new query update is handled by feedback.py and save binary for qureyHandle of queryOut.

Beside, I use a stopword frequency cutoff instead of stop word list. That is a will not count those term happened to in the files too often.

After parameter tuning, I finally use queryOut to output query result of top100 list in designated format.

Experiment

The experiment needs a lot of parameter tuning from BM25 k,b, stopword cutoff, tfidfmodel-selection, query type weight and combination.

After a period of trying, I setup the stopword cutoff at 20000 doc(neglect term in 20000 files) k1=1.8,=k3=2.4, b=0.75, query type equally added. The result showed a 0.74 in score

The stopword cutoff is very important in this case. For example if I set it to 40000 only 18 terms will be neglect but the score would drop below 0.7 and also a too low setting such as 10000 would lower the score.

In the other hand the weight of query type is not very useful. In the end a equal added result is the best.

I also test the model of simple tfidf, but the score is close to baseline. After that I only consider the BM25 model

I found the k1, k2 is also important a bad tuning would decrease the score.

The feedback only works for a query consulted before. That is it could only worked with the train query since I use train answer as feedback. The result improved slightly since I the tuning parameter I choose is very small. But this is important since we could not ask user to feedback all document at a time.

DIsccusion

I have learned a lot in this program. From the binary file saving, sparse matrix, utf-8 encoding to the basic way of IR query system. Although the parameter tuning part is time spending, I only test for a limited combination for this program. It requires more time and effort if I want to improve the result. It would be very helpful in further IR learning.