Konrad Rauscher
Information Retrieval
Project 1
12/08/16

Status:
Completed

hours: 25

Things I wish I had been told prior to being given assignment:
NA

Instructions

Implemented this assignment with my best classier and index: BM25 with stem. These are the only option for both classifier and index.

build.py Builds the index, accepts 3 arguments

1) [trec-files-directory-path] the directory containing the raw documents (e.g. fr940104.0)

2) [index-type] **This should be 'stem'**

3) [output-dir] the directory where index and lexicon files will be written

expan_reduc.py

Arguments (same as project two, except for 6) and 7), which are new):

1) [index-directory-path]
2) [query-file-path]
3) **[retrieval-mode]** *should be* 'bm25'
4) **[index-type]** *should be* 'stem'
5) [results-file-path]
6) [trec-files-directory-path]
7) **[query-type]** 'expand', 'filter', or 'expand-filter'

example:
expan_reduc.py
/tmp/my-indexes/
/Users/Konrad/Desktop/Mini-Trec-Data/QueryFile/queryfile.txt
bm25
stem
/Users/Konrad/Desktop/Project3_K_Rauscher/results-bm25-stem.txt
/Users/Konrad/Desktop/Mini-Trec-Data/BigSample/
expand-filter

Design

build.py
modified so that creates a mapping of docIDs to their respective bodies of text, which is used for query expansion.

Expansion
Strategy: Pseudo Relevance

Implementation:
1) obtain top *N* documents using BM25
2) load dictionary from file that maps docIDs to their respective texts (created from index creation)
3) for each query in the list of queries extracted from the query file,
    1) using the dictionary that maps docIDs to their respective texts, create a new dictionary that maps the the docIDs for the top *N* documents to their respective texts, which is used to create a list of all terms existing in the top *N* documents.
    2) using this list of the top *N* documents, for each term in this list:
        1) if term is in my index, and the term has not already been scored, assign a score to the term which is equal to the term's idf multiplied by the number of times the term occurs in the relevant set (the set of texts corresponding to the top *N* documents).
        2) append each term and its corresponding score to a list of such tuples.
    3) sort this list, and append the top *T* terms to the original query
4) run BM25 evaluation again, this time with the expanded terms

Reduction
Strategy: query threshold

Implementation:
made sure that the best term would always be added to list, regardless of threshold
1) obtained long query (Narrative) for each query instance.
2) for each query in the list of queries,
    1) for in term in a query,
        1) if term is in index, calculate idf
        2) if idf is greater than current max, update current max idf and current best term
        3) if term's idf is greater than the determined threshold, add to the list that comprises to the reduced query.
    2) if no terms have been added to the reduced query (is empty) then add the best term (which has been kept track of) to the reduced query, so that reduced queries are never empty, regardless of the threshold.
3) run BM25 evaluation on this list of reduced queries.

Expansion & Reduction
Strategy: Pseudo Relevance + query threshold
1) using short (Title) queries, queries are expanded using the above method.
2) the queries are then reduced using the above reduction method.

Experimental Plan

runs needed:

- expansion: adjusting T, N
- reduction: adjusting idf threshold
- expansion and reduction:
  - using best T and N, and adjusting idf
  - using best N, experiment with large T and adjusting idf

Report & Analysis

T = number of query expansion terms to add to original query
N = number of top documents to obtain potential expansion terms from

expansion

	T = 0	T = 1	T = 2	T = 3	T = 4	T = 5	…	T = 9	T = 10	T = 11	T = 12	T = 13
N = 0	0.2844						…					
N = 1		0.2721	0.2655	0.2599	0.2743	0.2731	…		0.2625	0.2692	0.2750	0.2730
N = 2		0.2977	0.2893	0.2831	0.3038	0.3015	…	0.3145	0.3290	0.3334	0.3342	0.3294
N = 3		0.2433	0.2303	0.2501	0.2348	0.2387	…		0.2718	0.2576	0.2624	0.2635
N = 4		0.2454	0.2529	0.2475	0.2383	0.2449	…		0.2389	0.2672	0.2659	0.2676
N = 5		0.2445	0.2527	0.2450	0.2435	0.2541	…		0.2423			
…		…	…	…	…	…	…		…			…
N = 10		0.2223	0.2287	0.2249	0.2371	0.2472	…		0.2176			0.2378

(scores are MAP)

The best map score occurs at N = 2 and T = 12 with a map score of 0.3342. This is a considerable increase from 0.2844, the map without query expansion.

One explanation for the observation that an N of one greater or less than 2 results in universally lower MAP scores is that the 2nd ranked document in the results from BM25 is relevant or particularly relevant for a majority of the queries, and that the 1st and 3rd ranked documents across all queries are less relevant as a whole in relation to 2nd documents as a whole. Thus, when increasing N from 1 to 2, the usefulness (which can be generally construed as uniqueness and relevancy) of expansion terms is greater as a whole, and when increasing N from 2 to 3, the particularly useful terms from the 2nd documents are being drowned out with less useful terms.

As one would expect, map begins to decrease quite markedly after a maximum or set or relatively adjacent maximums (in this case only N = 2 is the max) as the incorporation of more as well as less relevant documents begins to result in topic drift.

By using such a low N, query bias is a concern as using only the top 2 documents can lead to a form of overfitting as the expansion terms can end up being overly specific to these top two documents, rather than approximating the concept or topic represented by a given query.

It is worth noting that the tuning of these parameters is particularly specific to my BM25 classifier with this dataset. For example, the significantly better performance of N = 2 over any other N has almost certainly more to do with a significantly higher proportion of the 2nd ranked documents for all queries being relevant, rather than N = 2 being a generally good value to use with other classifiers or with a different data set.

No particular pattern or peculiarity of interest seems to exists in regard to the value chosen for T. The affect on given map scores seems highly dependent on the N involved. As such, T appears to be more of a tuning metric for N, rather than an independent tuning metric. This is what one would expect given the actual relationship that N and T share.

As expected, an examination of the actual terms being chosen to expand with observed terms that are relevant to the query. For example, the query ['herbal', 'food', 'supplement'] resulted in such expansion terms as: ['fda', 'food', 'prepared', 'potassium', 'sodium', 'hydroxypropyl', 'supplement', 'acid'].

reduction

IDF THRESHOLD	0.001	0.01	0.033	0.066	0.1	1
MAP	0.3466	0.3292	0.2715	0.2679	0.1943	0.0629

With a idf threshold of .001, approximately no more than 1 term is being removed from each long query (stop words have already been removed). As such, lower thresholds have no significant impact on map scores. At an idf threshold of 1, 17/20 queries comprised of only one term.

The distinct decrease in map scores with the raising of the idf threshold indicates that the majority of terms contained in each long query was relevant or useful in obtaining relevant documents. Because of this, and that stop words are already removed before applying reduction, the observation that raising the idf threshold always reduces map scores is reasonable.

expand-filter

IDF THRESHOLD	0.001	0.01	0.033	0.066	0.1	1
N	2	2	2	2	2	2
T	12	12	12	12	12	12
MAP	0.3135	0.3079	0.2756	0.2404	0.2213	0.1613

table 1

- Kept N = 2 as this value constantly provided the best map scores in expansion
- Tried a T = 12 as this T value, combined with an N of 2, provided the best map score.
- Varied the idf threshold from .001 to 1, as no change occurs to the map score outside this range.

Observations:
The same pattern seen in reduction analysis repeats itself here: all increases in the idf threshold reduce map scores.

I selected the parameters for this table to see if reduction could improve my best result from pure expansion. The lack of observed improvement points to all expansion terms being useful / helpful and that removing any of them constantly results in a worse map score.

IDF THRESHOLD	0.001	0.01	0.033	0.066	0.1	1
N	2	2	2	2	2	2
T	30	30	30	30	30	30
MAP	0.2994	0.2975	0.2812	0.3342	0.2346	0.1679

table 2. a

N	2
T	30
MAP (EXPANSION ONLY)	0.3002

table 2. b

Selected the parameters for these two tables to ascertain whether applying query reduction after doing expansion with a sub-optimally large number of expansion terms could lead to desirable results.

Again, however, the same pattern seen in reduction analysis repeats itself here: all increases in the idf threshold reduce map scores. Further, when compared to expansion only, (table 2. b) all expansion + reduction map scores are worse then with no reduction, holding N and T constant.

The distinct decrease in map scores with the raising of the idf threshold indicates that the majority of terms contained in each long query was relevant or useful in obtaining relevant documents. Because of this, and that stop words are already removed before applying reduction, the observation that raising the idf threshold always reduces map scores seems to be reasonable.