Question 1

The result after running query_boolean.py:

Index length: 808777

Workbooks

./gov/documents\14\G00-14-2198849

./gov/documents\36\G00-36-2337608

./gov/documents\50\G00-50-0062475

./gov/documents\69\G00-69-1400565

./gov/documents\69\G00-69-2624147

Australasia OR Airbase

./gov/documents\21\G00-21-0639911

./gov/documents\35\G00-35-2345443

./gov/documents\92\G00-92-1917825

./gov/documents\92\G00-92-3147280

Warm AND WELCOMING

./gov/documents\63\G00-63-2944034

./gov/documents\97\G00-97-2878104

Global AND SPACE AND economies

./gov/documents\26\G00-26-1275394

./gov/documents\42\G00-42-0794402

./gov/documents\56\G00-56-3884873

./gov/documents\60\G00-60-1685015

./gov/documents\70\G00-70-0584855

./gov/documents\70\G00-70-2258666

./gov/documents\84\G00-84-0274223

./gov/documents\96\G00-96-2539022

SCIENCE OR technology AND advancement AND PLATFORM

./gov/documents\29\G00-29-3700059

./gov/documents\94\G00-94-0708359

./gov/documents\99\G00-99-0289351

Wireless OR Communication AND channels OR SENSORY AND INTELLIGENCE

./gov/documents\33\G00-33-2857182

./gov/documents\49\G00-49-0055139

./gov/documents\65\G00-65-2419666

./gov/documents\68\G00-68-4089689

./gov/documents\84\G00-84-1559810

Question 2

Result after running query_tfidf.py:

Query: Is nuclear power plant eco-friendly?

Top-5 documents (similarity scores):

./gov/documents\76\G00-76-3273936 0.5802

./gov/documents\30\G00-30-1518511 0.4348

./gov/documents\01\G00-01-1806077 0.4302

./gov/documents\55\G00-55-0690938 0.4136

./gov/documents\72\G00-72-1085257 0.3936

Query: How to stay safe during severe weather?

Top-5 documents (similarity scores):

./gov/documents\99\G00-99-3847208 0.2812

./gov/documents\38\G00-38-1132272 0.1961

./gov/documents\75\G00-75-0577710 0.1534

./gov/documents\11\G00-11-0266936 0.1382

./gov/documents\66\G00-66-4088870 0.1356

Process finished with exit code 0

Result after running evaluate.py:

map: 0.1990

Rprec: 0.1675

recip_rank: 0.3867

P_5: 0.1733

P_10: 0.1467

P_20: 0.1000

Q3

Operation for process_tokens_1

What modifications you made?

After lowercase the word, I also stem the word using PorterStemmer.

Why you made them?

Because stemming can do some crude heuristic process that strips off suffixes. This can help make the result more accurate in the process of building the dictionary (index here) to help us get more accurate results after querying.

What the new performance is?

Result for process_tokens_1(toks):

map: 0.2343

Rprec: 0.2095

recip_rank: 0.4135

P_5: 0.2133

P_10: 0.1600

P_20: 0.1100

Time for running indexer: 4:37:70

Why you think the modification did/did not work?

It takes lots of time to get the result. But the accuracy is improved obviously. This contribution should belong to the crude heuristic process that strips off suffixes of stemming. For example, for the word “workbooks”, before we stem this word, there are two words “workbooks” and “workbook” in the dictionary, but they actually represent the same thing. The documents they belong to should be relevant and returned if a person queries the word “workbook” or “workbooks”. After stemming, this can be fixed because there is only “workbook” in the dictionary and the result is more precise because we strip off suffixes.

Operation for process_tokens_2

What modifications you made?

After lowercase the word, I also do lemmatization for the word using WordNetLemmatizer.

Why you made them?

Because lemmatization can help to turn terms into words that can be looked up in the dictionary, which can be more accurate. So, we can get more accurate dictionary(index) so that we can get more accurate results when we query.

What the new performance is?

Result for process_tokens_2(toks):

map: 0.2517

Rprec: 0.2507

recip_rank: 0.4625

P_5: 0.2400

P_10: 0.1700

P_20: 0.1183

Time: 1:21:58

Why you think the modification did/did not work?

The result is better than the original one but not very obviously better compared to result after stemming. I think we did get a more accurate dictionary here and this can improve the performance indeed. But the dictionary is too accurate. For example, if we query “run” in our collection, “running” and “run” indeed can be transferred to “run” after lemmatization and the relevant documents can be returned, but for “runner”, it will be kept to be “runner”, so the relevant documents which contain “runner” will not be returned, which can make the result less precise.

Operation for process_tokens_3

What modifications you made?

After lowercase the word, I also remove some special symbols included in term like comma, question mark and period mark.

Why you made them?

Because I notice that there are some term contains these special symbols, which can make the dictionary less accurate.

What the new performance is?

Result for process_tokens_3(toks):

map: 0.2344

Rprec: 0.2287

recip_rank: 0.4588

P_5: 0.2067

P_10: 0.1733

P_20: 0.1150

Time: 54.49

Why you think the modification did/did not work?

The result is better than the original one. Like what I have mentioned before, it's because the terms with special symbols are less in the dictionary(index). For example, term "workbooks..." now has become "workbooks", which can help to get more accurate result when we are querying.

Operation for process_tokens_4

What modifications you made?

After lowercase the word, I also do lemmatization for the word using WordNetLemmatizer and then I remove some special symbols except letters and numbers included in term.

Why you made them?

Because I think that I can get more accurate result after these two steps than when they are made solely.

What the new performance is?

Result for process_tokens_4(toks):

map: 0.2654

Rprec: 0.2370

recip_rank: 0.4897

P_5: 0.2600

P_10: 0.1900

P_20: 0.1217

Time: 1:33:97

Why you think the modification did/did not work?

We did get better results than we did each of them solely. This is because we also restored the word on the basis of removing special symbols. Besides, although lemmatization can help

to remove punctuation marks, it cannot remove some special symbols like # or @. For example, for the term "workbook.#@s", we can get "workbook" in the end, which is better for the precision of the query.

Comparison of four methods:

Table 1: Results for the four processes

Process Method	Time (minutes : seconds : milliseconds)	MAP	Rprec	recip_rank	P-20
Stemming	4:37:70	0.2343	0.2095	0.4135	0.1100
Lemmatization	1:21:58	0.2517	0.2507	0.4625	0.1183
Special symbol removal	0:54:49	0.2344	0.2287	0.4588	0.1150
Lemmatization and special symbol removal	1:33:97	0.2654	0.2370	0.4897	0.1217

From Table 1, we can see that there are some indexes that we can use to compare different process methods. The different index focuses on various aspects of the performance of the system. To determine which process is the best, we need to know the reason why people or our clients need the query system. For example, if our clients mainly think highly of the processing time, then the "special symbol removal" process is the best for them.

However, the "Lemmatization and special symbol removal" process is the best of the four processes above overall.

First, its MAP is the highest, which means we get higher average precision of the overall average precision for each query, this represents that this process can make the query system more precise wholely.

Then, it has good Rprec, which represents the proportion of relevant documents in the first R returned results is higher, we can get better results in the first few results after querying. Although "Lemmatization" is better than it in this aspect, "Lemmatization" can't catch up with it in other aspects.

Next, it has the best recip_rank, from which we can say that the ranking of the first occurrence of relevant documents in the search system's returned results is higher, we can get the first relevant result faster.

Besides, P_5, P_10, and P-20 (only P_20 is shown in the chart) are the highest. This indicates that the fraction of retrieved documents that are relevant is higher, and we can get more relevant results.

Last, the time it consumes is still acceptable, which is close to ""Lemmatization".

All in all, although the "Lemmatization and special symbol removal" process is the best overall, it could not always be the best everywhere. The most important issue is to understand what exactly our clients want and judge which one is the best according to the indexes above. The suitable one is always the best one.