Assignment Two Report

Student ID: s3300154 Name: Ling Liu E-mail: s3300154@student.rmit.edu.au

1. Ranked Retrieval

The implementation of **BM25 similarity function** mainly includes two programs: **index.java** and **search.java**. The main modification for **index.java** program is to add the function for calculating document weight(**document length**) at indexing. And **Search.java** program implements using accumulators to calculate the document similarity score for each document and using a min-heap data structure to select the top documents with the highest similarity scores.

The first task is to calculate the document length for each document at indexing time. First of all, I will introduce a new class **DocumentNode.java** which includes: original document number, document length, document start position and document end position which records the content of each document position (it records the line number range between <TEXT> and </TEXT> tag). In this assignment, I just use the first two components to store original document number and document length. I don't use the last two components for my solution, but I still keep them, we may need it in future. Now I will talk about **how to calculate the document length**. In my algorithms, I will count the total number of all the characters which include all the letters, number and space as document length. For example, if document content is "This is my NO.1 document!", the length of the document is: 24 that means this document includes 24 characters(**One thing has to be mention** here is before we start counting the document length, the program will perform filter operation which includes remove all punctuation operation). The program will store document length in **map file**.

The second task is how to use accumulators to accrue partial similarity scores as each term is processed and to use min-heap data structure to select the top documents with the highest scores. The **MinHeapOrderAlgorithm.java** program implements the min-heap algorithm to select the top-n documents. This algorithm follows the heap sort algorithm idea. I don't explain detail. Here I just mention **DocScoreNode.java** class which contains three elements in it: document number, document similarity score and document description. The heap order program will apply order operation on these objects. Now let's start to introduce how to calculate the document similarity score. According to the BM25 formula, we need **following components** to calculate document score: **document length** which already know at indexing time, **average document length** which can calculate by sum(all the document length)/the number of documents in the collection, **ft** which is the number of documents containing term t, **fdt** which is the number of occurrences of t in document(Document length information is stored in **map file** and ft-fdt information are stored in **invlists file**). So when the search.java program starts, it will perform follow operation:

- Read **lexicon file** which contains term-document frequency information on disk and **map file** which contains document number and length information into memory. The program will also calculate the average length of document when reading the map file.
- Filter the input query with the same rule as at indexing time.
- Process query, one term at a time. The program will fetch term lists from invlists file on disk by matching the lexicon file. According to BM25 formula, it will use accumulators to accrue partial similarity scores as each term processed. Then it will sum individual term score to calculate the final document similarity score for each document.
- When program calculates the score for each document, it will call **MinHeapOrderAlgorithm** to select the top n documents with the highest similarity scores. This program just can select the top n documents from the whole collection, it can not perform sort operation. So the program will use a **TreeMap** structure to sort the top n documents by descending order.

- The last step is that the program will display the top n documents and compare these retrieved top n document with relevance judgements **qrels file** and print the common documents which they both contain.

2. Query Expansion

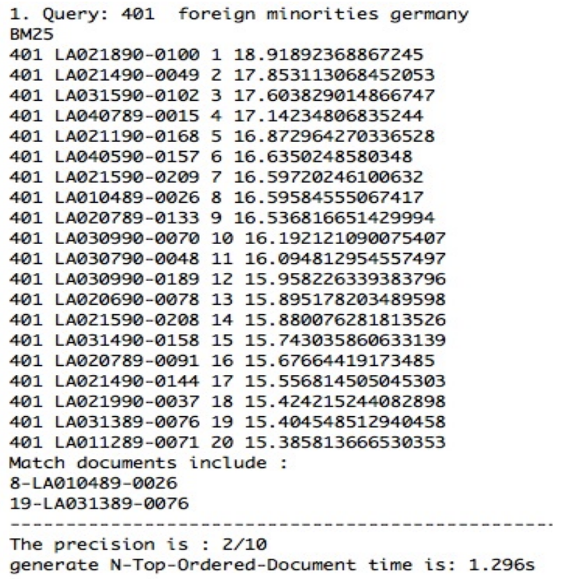
The implementation of **Query Expansion** function mainly includes two programs: **index.java** and **SearchQueryExpansion.java**. The main task of **index.java** program for this part is to control the whole process of Query Expansion operation. It will **reuse** most functions of the BM25 algorithm. The major work of Query Expansion is implemented by **SearchQueryExpansion.java** program. It will calculate weight for all terms which are contained in the top set collection and select the top E terms for query expansion.

Here I will explain Query Expansion operation step by step:

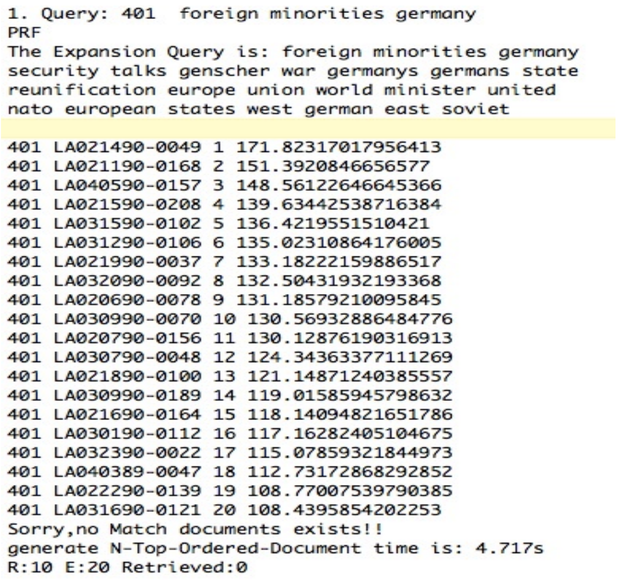
- User input parameters for query expansion from console (n, R, E, Query String).
- According to the R which specifies the number of top-ranked documents that will be used for expansion, the **search.java** program will evaluate the initial query by perform the same operation as BM25 algorithm to get these R-top-ranked documents.
- When program gets the expansion documents collection from last step, it will call **generateWordsExpansionCollection method** from **SearchQueryExpansion.java** class to fetch the top R documents from disk. There are two different ways to fetch documents from disk. **The one is**: read expansion documents collection from **collection file** on disk, generate terms collection from these documents, calculate term weight for each terms in collection. **The other is**: scan **invlists file** line by line directly; check whether the document number is included in expansion documents collection, if yes, that means the current term is included in expansion documents collection and the program will calculate the term weight(**note:** in order to improve the performance, the program will **record** the **largest document number** in expansion documents collection, it will be used to **compare** this number with the current term document occurrence number, if document occurrence number is larger than this number, the program will jump to deal with next term). When this scanning process finish, the program will accumulate partial score to obtain the term final weight. In this assignment, the programs are implemented by the second method.
- From the last step the program stores all the term score into memory, then we can call **MinHeapOrderAlgorithm** to select the top E terms with the highest term weight. **Two rules** have to be mentioned here: **the one** is that if program finds any term is the same as term included in input query, the program will ignore this term and fetch next term instead of repeated term. **The other** is that the program will check where the current term is included in **stoplist_sorted file ,** if yes that means this term is a stop word and the program will ignore this term and keep checking next term. If the term is not satisfied these two rules, it will be selected as expansion term. Now we get the Top-E terms from the top R expansion documents.
- The next step is append top-E terms which generate from last steps to original query and perform our ranked retrieval algorithm again to obtain the final top-n-ranked documents collection.
- The last step is display the top n documents and compare these retrieved top n document with relevance judgements **qrels file** and print the common documents which they both contain.

3. Evaluation

All the answers lists show below:

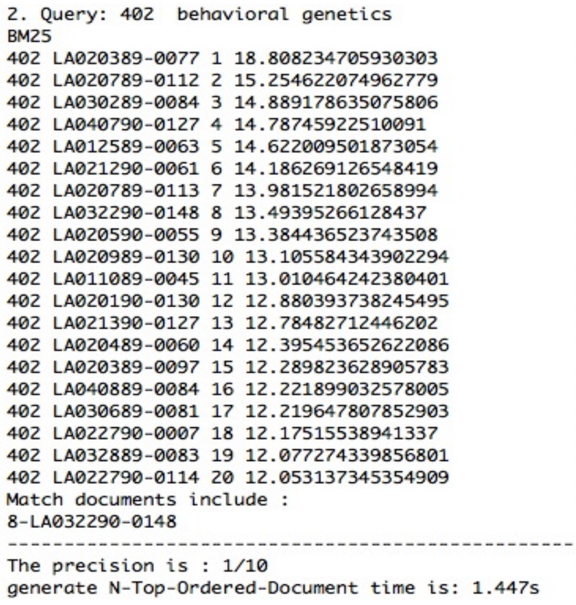


```
1. Query: 401  foreign minorities germany
BM25
401 LA021890-0100 1 18.91892368867245
401 LA021490-0049 2 17.853113068452053
401 LA031590-0102 3 17.603829014866747
401 LA040789-0015 4 17.14234806835244
401 LA021190-0168 5 16.872964270336528
401 LA040590-0157 6 16.6350248580348
401 LA021590-0209 7 16.59720246100632
401 LA010489-0026 8 16.59584555067417
401 LA020789-0133 9 16.536816651429994
401 LA030990-0070 10 16.192121090075407
401 LA030790-0048 11 16.094812954557497
401 LA030990-0189 12 15.958226339383796
401 LA020690-0078 13 15.895178203489598
401 LA021590-0208 14 15.880076281813526
401 LA031490-0158 15 15.743035860633139
401 LA020789-0091 16 15.67664419173485
401 LA021490-0144 17 15.556814505045303
401 LA021990-0037 18 15.424215244082898
401 LA031389-0076 19 15.404548512940458
401 LA011289-0071 20 15.385813666530353
Match documents include :
8-LA010489-0026
19-LA031389-0076
-----------------------------------------------
The precision is : 2/10
generate N-Top-Ordered-Document time is: 1.296s
```

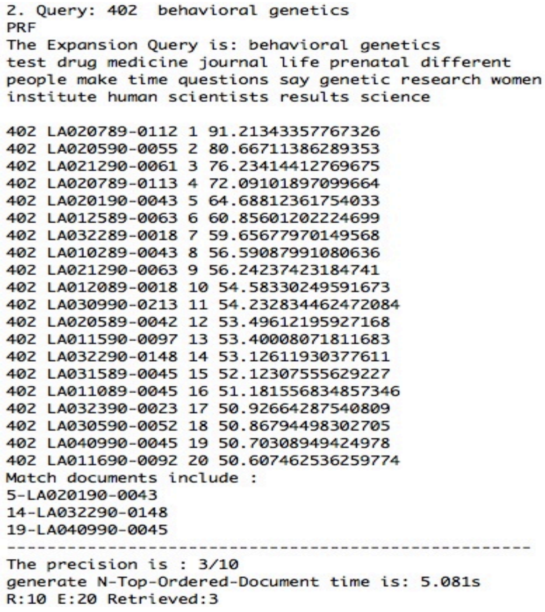


```
1. Query: 401  foreign minorities germany
PRF
The Expansion Query is: foreign minorities germany
security talks genscher war germanys germans state
reunification europe union world minister united
nato european states west german east soviet

401 LA021490-0049 1 171.82317017956413
401 LA021190-0168 2 151.3920846656577
401 LA040590-0157 3 148.56122646645366
401 LA021590-0208 4 139.63442538716384
401 LA031590-0102 5 136.4219551510421
401 LA031290-0106 6 135.02310864176005
401 LA021990-0037 7 133.18222159886517
401 LA032090-0092 8 132.50431932193368
401 LA020690-0078 9 131.18579210095845
401 LA030990-0070 10 130.56932886484776
401 LA020790-0156 11 130.12876190316913
401 LA030790-0048 12 124.34363377111269
401 LA021890-0100 13 121.14871240385557
401 LA030990-0189 14 119.01585945798632
401 LA021690-0164 15 118.14094821651786
401 LA030190-0112 16 117.16282405104675
401 LA032390-0022 17 115.07859321844973
401 LA040389-0047 18 112.73172868292852
401 LA022290-0139 19 108.77007539790385
401 LA031690-0121 20 108.4395854202253
Sorry,no Match documents exists!!
generate N-Top-Ordered-Document time is: 4.717s
R:10 E:20 Retrieved:0
```

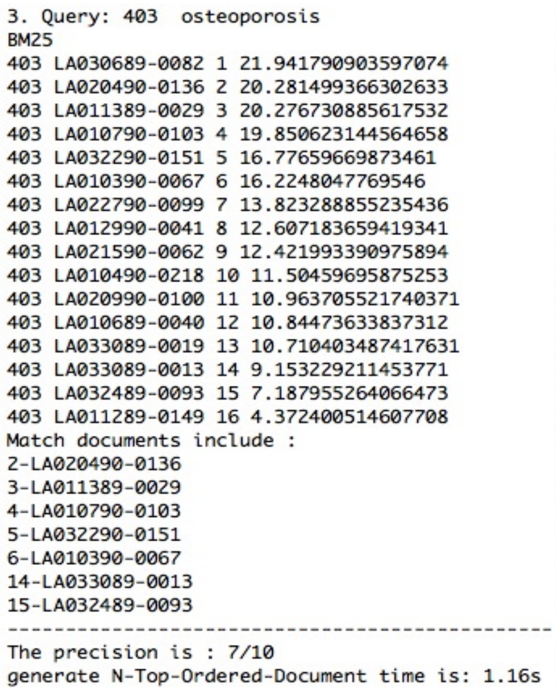


```
2. Query: 402  behavioral genetics
BM25
402 LA020389-0077 1 18.808234705930303
402 LA020789-0112 2 15.254622074962779
402 LA030289-0084 3 14.889178635075806
402 LA040790-0127 4 14.78745922510091
402 LA012589-0063 5 14.622009501873054
402 LA021290-0061 6 14.186269126548419
402 LA020789-0113 7 13.981521802658994
402 LA032290-0148 8 13.49395266128437
402 LA020590-0055 9 13.384436523743508
402 LA020989-0130 10 13.105584343902294
402 LA011089-0045 11 13.010464242380401
402 LA020190-0130 12 12.880393738245495
402 LA021390-0127 13 12.78482712446202
402 LA020489-0060 14 12.395453652622086
402 LA020389-0097 15 12.289823628905783
402 LA040889-0084 16 12.221899032578005
402 LA030689-0081 17 12.219647807852903
402 LA022790-0007 18 12.17515538941337
402 LA032889-0083 19 12.077274339856801
402 LA022790-0114 20 12.053137345354909
Match documents include :
8-LA032290-0148
-----------------------------------------------
The precision is : 1/10
generate N-Top-Ordered-Document time is: 1.447s
```

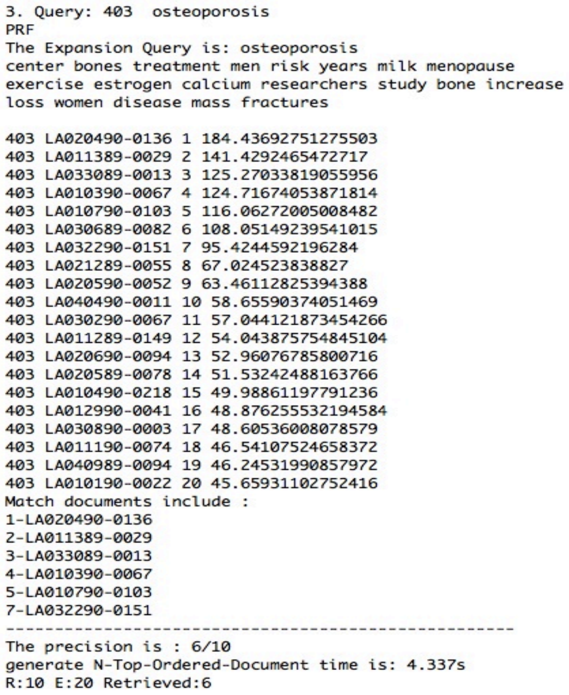


```
2. Query: 402  behavioral genetics
PRF
The Expansion Query is: behavioral genetics
test drug medicine journal life prenatal different
people make time questions say genetic research women
institute human scientists results science

402 LA020789-0112 1 91.21343357767326
402 LA020590-0055 2 80.66711386289353
402 LA021290-0061 3 76.23414412769675
402 LA020789-0113 4 72.09101897099664
402 LA020190-0043 5 64.68812361754033
402 LA012589-0063 6 60.85601202224699
402 LA032289-0018 7 59.65677970149568
402 LA010289-0043 8 56.59087991080636
402 LA021290-0063 9 56.24237423184741
402 LA012089-0018 10 54.58330249591673
402 LA030990-0213 11 54.232834462472084
402 LA020589-0042 12 53.49612195927168
402 LA011590-0097 13 53.40008071811683
402 LA032290-0148 14 53.12611930377611
402 LA031589-0045 15 52.12307555629227
402 LA011089-0045 16 51.181556834857346
402 LA032390-0023 17 50.92664287540809
402 LA030590-0052 18 50.86794498302705
402 LA040990-0045 19 50.70308949424978
402 LA011690-0092 20 50.607462536259774
Match documents include :
5-LA020190-0043
14-LA032290-0148
19-LA040990-0045
-----------------------------------------------
The precision is : 3/10
generate N-Top-Ordered-Document time is: 5.081s
R:10 E:20 Retrieved:3
```

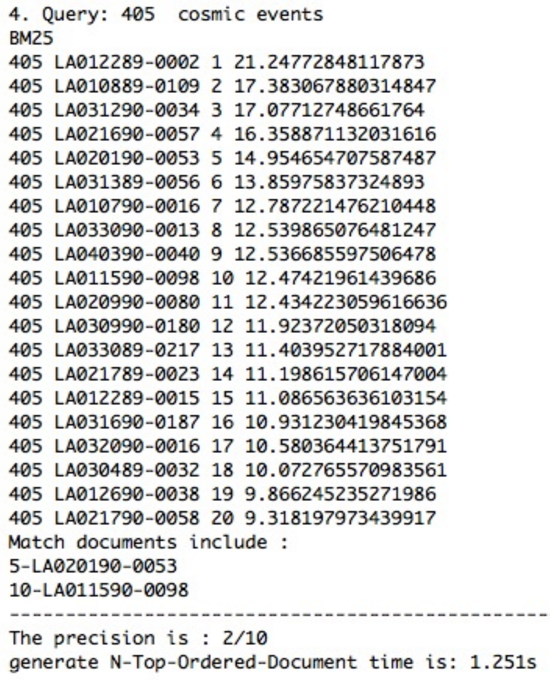


```
3. Query: 403  osteoporosis
BM25
403 LA030689-0082 1 21.941790903597074
403 LA020490-0136 2 20.281499366302633
403 LA011389-0029 3 20.276730885617532
403 LA010790-0103 4 19.850623144564658
403 LA032290-0151 5 16.77659669873461
403 LA010390-0067 6 16.2248047769546
403 LA022790-0099 7 13.823288855235436
403 LA012990-0041 8 12.607183659419341
403 LA021590-0062 9 12.421993390975894
403 LA010490-0218 10 11.50459695875253
403 LA020990-0100 11 10.963705521740371
403 LA010689-0040 12 10.84473633837312
403 LA033089-0019 13 10.710403487417631
403 LA033089-0013 14 9.153229211453771
403 LA032489-0093 15 7.187955264066473
403 LA011289-0149 16 4.372400514607708
Match documents include :
2-LA020490-0136
3-LA011389-0029
4-LA010790-0103
5-LA032290-0151
6-LA010390-0067
14-LA033089-0013
15-LA032489-0093
-----------------------------------------------
The precision is : 7/10
generate N-Top-Ordered-Document time is: 1.16s
```

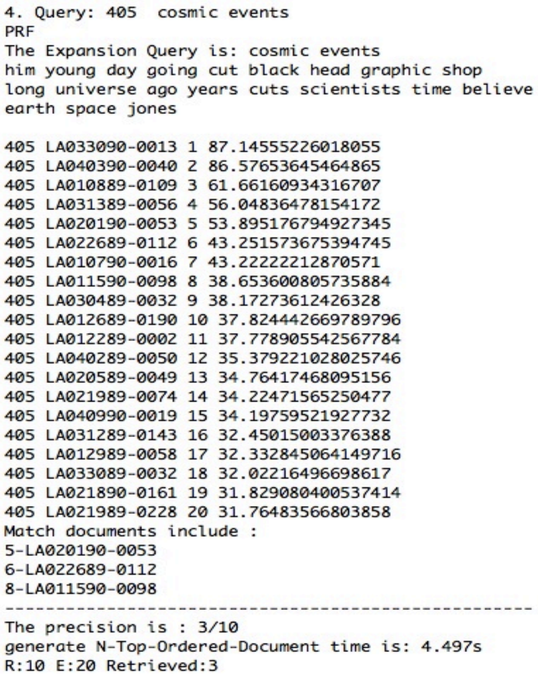


```
3. Query: 403  osteoporosis
PRF
The Expansion Query is: osteoporosis
center bones treatment men risk years milk menopause
exercise estrogen calcium researchers study bone increase
loss women disease mass fractures

403 LA020490-0136 1 184.43692751275503
403 LA011389-0029 2 141.4292465472717
403 LA033089-0013 3 125.27033819055956
403 LA010390-0067 4 124.71674053871814
403 LA010790-0103 5 116.06272005008482
403 LA030689-0082 6 108.05149239541015
403 LA032290-0151 7 95.4244592196284
403 LA021289-0055 8 67.024523838827
403 LA020590-0052 9 63.46112825394388
403 LA040490-0011 10 58.65590374051469
403 LA030290-0067 11 57.044121873454266
403 LA011289-0149 12 54.043875754845104
403 LA020690-0094 13 52.96076785800716
403 LA020589-0078 14 51.53242488163766
403 LA010490-0218 15 49.98861197791236
403 LA012990-0041 16 48.876255532194584
403 LA030890-0003 17 48.60536008078579
403 LA011190-0074 18 46.54107524658372
403 LA040989-0094 19 46.24531990857972
403 LA010190-0022 20 45.65931102752416
Match documents include :
1-LA020490-0136
2-LA011389-0029
3-LA033089-0013
4-LA010390-0067
5-LA010790-0103
7-LA032290-0151
-----------------------------------------------
The precision is : 6/10
generate N-Top-Ordered-Document time is: 4.337s
R:10 E:20 Retrieved:6
```

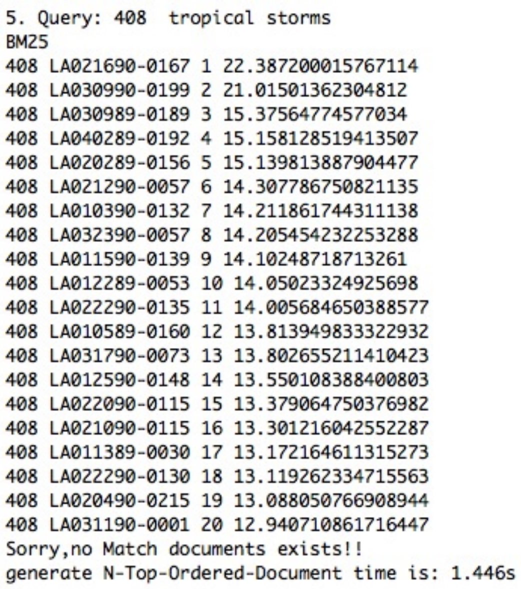


```
4. Query: 405  cosmic events
BM25
405 LA012289-0002 1 21.24772848117873
405 LA010889-0109 2 17.383067880314847
405 LA031290-0034 3 17.07712748661764
405 LA021690-0057 4 16.358871132031616
405 LA020190-0053 5 14.954654707587487
405 LA031389-0056 6 13.85975837324893
405 LA010790-0016 7 12.787221476210448
405 LA033090-0013 8 12.539865076481247
405 LA040390-0040 9 12.536685597506478
405 LA011590-0098 10 12.47421961439686
405 LA020990-0080 11 12.434223059616636
405 LA030990-0180 12 11.92372050318094
405 LA033089-0217 13 11.403952717884001
405 LA021789-0023 14 11.198615706147004
405 LA012289-0015 15 11.086563636103154
405 LA031690-0187 16 10.931230419845368
405 LA032090-0016 17 10.580364413751791
405 LA030489-0032 18 10.072765570983561
405 LA012690-0038 19 9.866245235271986
405 LA021790-0058 20 9.318197973439917
Match documents include :
5-LA020190-0053
10-LA011590-0098
------------------------------------------------
The precision is : 2/10
generate N-Top-Ordered-Document time is: 1.251s
```

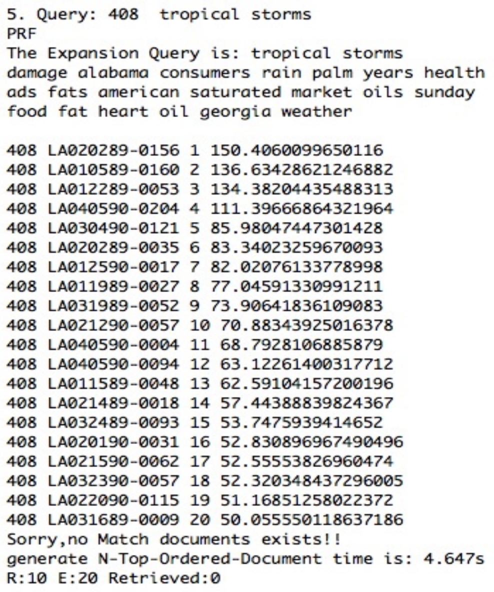


```
4. Query: 405  cosmic events
PRF
The Expansion Query is: cosmic events
him young day going cut black head graphic shop
long universe ago years cuts scientists time believe
earth space jones

405 LA033090-0013 1 87.14555226018055
405 LA040390-0040 2 86.57653645464865
405 LA010889-0109 3 61.66160934316707
405 LA031389-0056 4 56.04836478154172
405 LA020190-0053 5 53.895176794927345
405 LA022689-0112 6 43.251573675394745
405 LA010790-0016 7 43.22222212870571
405 LA011590-0098 8 38.653600805735884
405 LA030489-0032 9 38.17273612426328
405 LA012689-0190 10 37.824442669789796
405 LA012289-0002 11 37.778905542567784
405 LA040289-0050 12 35.379221028025746
405 LA020589-0049 13 34.76417468095156
405 LA021989-0074 14 34.22471565250477
405 LA040990-0019 15 34.19759521927732
405 LA031289-0143 16 32.45015003376388
405 LA012989-0058 17 32.332845064149716
405 LA033089-0032 18 32.02216496698617
405 LA021890-0161 19 31.829080400537414
405 LA021989-0228 20 31.76483566803858
Match documents include :
5-LA020190-0053
6-LA022689-0112
8-LA011590-0098
----------------------------------------------------
The precision is : 3/10
generate N-Top-Ordered-Document time is: 4.497s
R:10 E:20 Retrieved:3
```



```
5. Query: 408  tropical storms
BM25
408 LA021690-0167 1 22.387200015767114
408 LA030990-0199 2 21.01501362304812
408 LA030989-0189 3 15.37564774577034
408 LA040289-0192 4 15.158128519413507
408 LA020289-0156 5 15.139813887904477
408 LA021290-0057 6 14.307786750821135
408 LA010390-0132 7 14.211861744311138
408 LA032390-0057 8 14.205454232253288
408 LA011590-0139 9 14.10248718713261
408 LA012289-0053 10 14.05023324925698
408 LA022290-0135 11 14.005684650388577
408 LA010589-0160 12 13.813949833322932
408 LA031790-0073 13 13.802655211410423
408 LA012590-0148 14 13.550108388400803
408 LA022090-0115 15 13.379064750376982
408 LA021090-0115 16 13.301216042552287
408 LA011389-0030 17 13.172164611315273
408 LA022290-0130 18 13.119262334715563
408 LA020490-0215 19 13.088050766908944
408 LA031190-0001 20 12.940710861716447
Sorry,no Match documents exists!!
generate N-Top-Ordered-Document time is: 1.446s
```



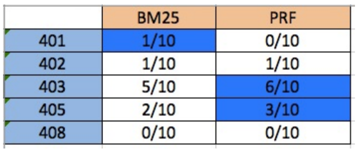
```
5. Query: 408  tropical storms
PRF
The Expansion Query is: tropical storms
damage alabama consumers rain palm years health
ads fats american saturated market oils sunday
food fat heart oil georgia weather

408 LA020289-0156 1 150.4060099650116
408 LA010589-0160 2 136.63428621246882
408 LA012289-0053 3 134.38204435488313
408 LA040590-0204 4 111.39666864321964
408 LA030490-0121 5 85.98047447301428
408 LA020289-0035 6 83.34023259670093
408 LA012590-0017 7 82.02076133778998
408 LA011989-0027 8 77.04591330991211
408 LA031989-0052 9 73.90641836109083
408 LA021290-0057 10 70.88343925016378
408 LA040590-0004 11 68.7928106885879
408 LA040590-0094 12 63.12261400317712
408 LA011589-0048 13 62.59104157200196
408 LA021489-0018 14 57.44388839824367
408 LA032489-0093 15 53.747593941465
408 LA020190-0031 16 52.830896967490496
408 LA021590-0062 17 52.55553826960474
408 LA032390-0057 18 52.320348437296005
408 LA022090-0115 19 51.16851258022372
408 LA031689-0009 20 50.055550118637186
Sorry,no Match documents exists!!
generate N-Top-Ordered-Document time is: 4.647s
R:10 E:20 Retrieved:0
```

Explanation: All the answer lists for each query are showed above. For each query, the **BM25** result lists show on **left side** and the **PRF** result lists show on the **right side**. The PRF result lists also display the **query expansion terms**. For each result lists, it also includes **document match** information and query **running time** information.

Note: One thing I have to mention here is that all these results are tested on my local machine, I have also done the same experiments on yallara. The difference between these two tests is the running time. When the same programs run on yallara, it needs about 5 times longer than the programs running on my local machine.

3.2 Compare the systems performance using P@10 metric



	BM25	PRF
401	1/10	0/10
402	1/10	1/10
403	5/10	6/10
405	2/10	3/10
408	0/10	0/10

The left side table show precision for each query as p@10 metric. For query 401, BM25 algorithm is better than PRF algorithm. For query 402,the BM25 and PRF algorithm obtain the same precision score. For query 403,405, PRF is better than BM25. But for query 408, both of the algorithm can not find any document in top 10 documents. By using P@10 as metric for these testing results, PRF looks better than BM25 algorithm.

Average Precision Results for these two algorithms:

BM25: (1/10+1/10+5/10+2/10+0/10)/5=0.18

PRF : (0/10+1/10+6/10+3/10+0/10)/5=0.2

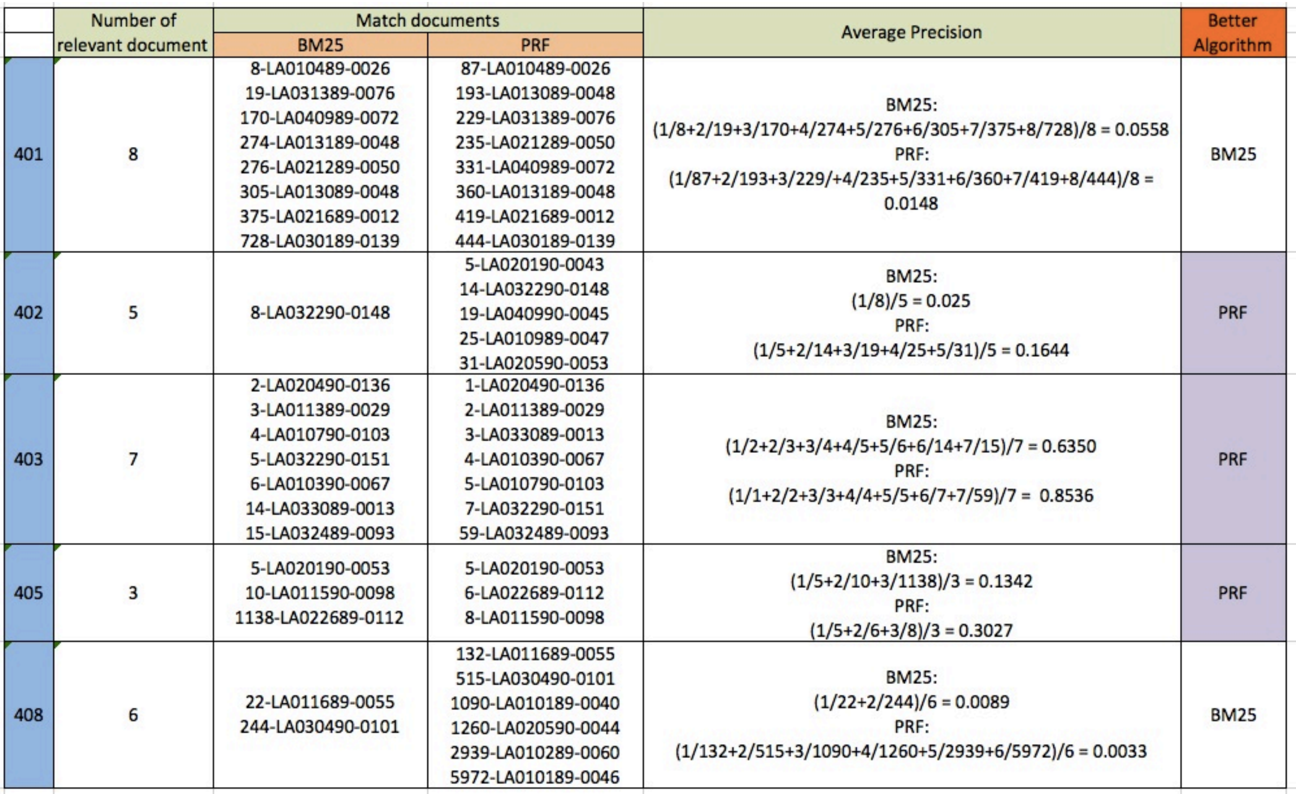
I think P@10 metric is not a sensible metric for comparing these two systems. The reasons are:

- The algorithm just considers the total number of the retrieved document in top 10 returned collection. But it doesn't consider the position of retrieved documents. In some situation, documents that are retrieved earlier in the ranking should contribute more to the score.
- p@10 is a popular metric since it reflects the first page performance for web search engine. It may not good for our collection data set.
- The last reason is that we have just tested five queries for this experiment. It is very hard to say which system is better than the other. One system may good for some specific queries whereas the other may good for others. This means that a set of test information needs must be large and diverse enough to be representative of system effectiveness across different queries.

3.3 Other Evaluation Metric

In this part, I will choose **mean average precision (MAP)** as our evaluation metric. It provides a single-figure measure of quality across recall levels. Among evaluation measures, MAP has been shown to have especially good discrimination and stability. For a single information need, average precision is the average of the precision value obtained for the set of top n documents existing after each relevant document is retrieved, and this value is then averaged over information needs. So I think MAP metric is appropriate for comparing the performance of these two systems.

In my experiment, I will choose following value for each parameter: **n=35472 R=10 E=20** and running all our queries. The test results show below:



	Number of	Match documents		Average Precision	Better
	relevant document	BM25	PRF		Algorithm
401	8	8-LA010489-0026 19-LA031389-0076 170-LA040989-0072 274-LA013189-0048 276-LA021289-0050 305-LA013089-0048 375-LA021689-0012 728-LA030189-0139	87-LA010489-0026 193-LA013089-0048 229-LA031389-0076 235-LA021289-0050 331-LA040989-0072 360-LA013189-0048 419-LA021689-0012 444-LA030189-0139	BM25: (1/8+2/19+3/170+4/274+5/276+6/305+7/375+8/728)/8 = 0.0558 PRF: (1/87+2/193+3/229/+4/235+5/331+6/360+7/419+8/444)/8 = 0.0148	BM25
402	5	8-LA032290-0148	5-LA020190-0043 14-LA032290-0148 19-LA040990-0045 25-LA010989-0047 31-LA020590-0053	BM25: (1/8)/5 = 0.025 PRF: (1/5+2/14+3/19+4/25+5/31)/5 = 0.1644	PRF
403	7	2-LA020490-0136 3-LA011389-0029 4-LA010790-0103 5-LA032290-0151 6-LA010390-0067 14-LA033089-0013 15-LA032489-0093	1-LA020490-0136 2-LA011389-0029 3-LA033089-0013 4-LA010390-0067 5-LA010790-0103 7-LA032290-0151 59-LA032489-0093	BM25: (1/2+2/3+3/4+4/5+5/6+6/14+7/15)/7 = 0.6350 PRF: (1/1+2/2+3/3+4/4+5/5+6/7+7/59)/7 = 0.8536	PRF
405	3	5-LA020190-0053 10-LA011590-0098 1138-LA022689-0112	5-LA020190-0053 6-LA022689-0112 8-LA011590-0098	BM25: (1/5+2/10+3/1138)/3 = 0.1342 PRF: (1/5+2/6+3/8)/3 = 0.3027	PRF
408	6	22-LA011689-0055 244-LA030490-0101	132-LA011689-0055 515-LA030490-0101 1090-LA010189-0040 1260-LA020590-0044 2939-LA010289-0060 5972-LA010189-0046	BM25: (1/22+2/244)/6 = 0.0089 PRF: (1/132+2/515+3/1090+4/1260+5/2939+6/5972)/6 = 0.0033	BM25

Explanation: The first column shows the query number. **The second column** shows the number of relevant document for each query. These numbers obtain from **qrels file.** I count all relevant documents for each query which the fourth item is “1”. (Note: In qrels file, we may find some documents are considered as relevant documents but we can not find these documents in our collection file. That’s because the highest document number in our collection file is LA041089-0095, but qrels file gives us some documents which are not included in our collection file, so we ignore these relevant documents). **The third and fourth column** shows the common documents from which compare with relevant documents collection for BM25 and PRF algorithms respectively. **The fifth column** shows the average precision score for each query. And **the last column** shows the better algorithm for each query.

According to this table, we can calculate the **MAP** as below:

BM25: (0.0558+0.025+0.6350+0.1342+0.0089)/5=0.171

PRF: (0.0148+0.1644+0.8536+0.3027+0.0033)/5=0.267

Conclusion: from this experiment, we can see PRF algorithm is better than BM25. This result is the same as P@10 metric.

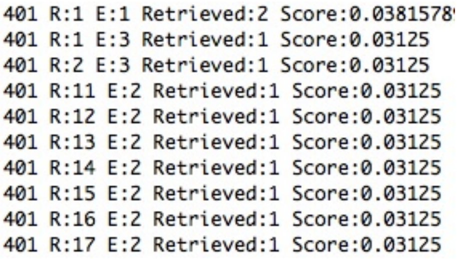
4. Optional Extension:

4.1 Query Expansion Parameters

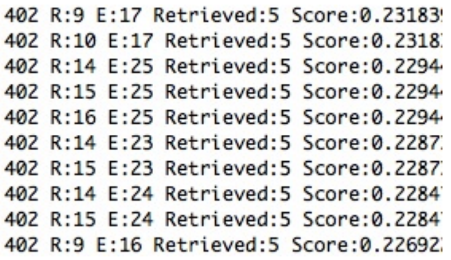
I have written a small program for this experiments. The code is included in **search.java** program from **line 608** to **line 681**. If you want to test this program just uncomment this part of program and run it. The running results file is included in my assignment submit directory named **REScoreFile.**

An example line from the file is: **401 R:1 E:3 Retrieved:1 Score:0.03125**

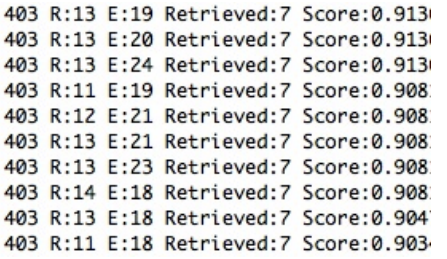
Where the **first item** is a query number, the **second and third item** are the value of R and E parameter respectively, the **fourth item** is the number of the match documents and the **last item** show the **Average Precision** for these set of parameters. The following figures show the top 10 highest score for each individual query (in this experiment the program sets the parameters as: n=30, R: from 1 to 20, E: from 1 to 25):



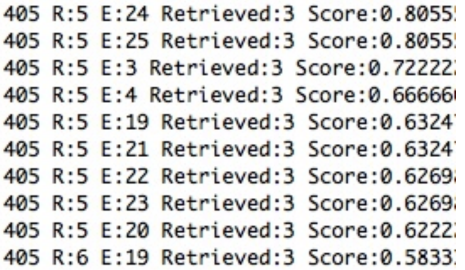
```
401 R:1 E:1 Retrieved:2 Score:0.0381578
401 R:1 E:3 Retrieved:1 Score:0.03125
401 R:2 E:3 Retrieved:1 Score:0.03125
401 R:11 E:2 Retrieved:1 Score:0.03125
401 R:12 E:2 Retrieved:1 Score:0.03125
401 R:13 E:2 Retrieved:1 Score:0.03125
401 R:14 E:2 Retrieved:1 Score:0.03125
401 R:15 E:2 Retrieved:1 Score:0.03125
401 R:16 E:2 Retrieved:1 Score:0.03125
401 R:17 E:2 Retrieved:1 Score:0.03125
```



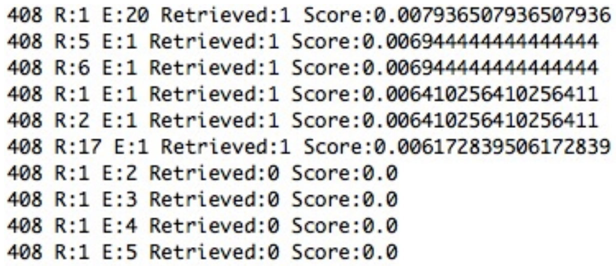
```
402 R:9 E:17 Retrieved:5 Score:0.23183
402 R:10 E:17 Retrieved:5 Score:0.2318
402 R:14 E:25 Retrieved:5 Score:0.2294
402 R:15 E:25 Retrieved:5 Score:0.2294
402 R:16 E:25 Retrieved:5 Score:0.2294
402 R:14 E:23 Retrieved:5 Score:0.2287
402 R:15 E:23 Retrieved:5 Score:0.2287
402 R:14 E:24 Retrieved:5 Score:0.2284
402 R:15 E:24 Retrieved:5 Score:0.2284
402 R:9 E:16 Retrieved:5 Score:0.22692
```



```
403 R:13 E:19 Retrieved:7 Score:0.913
403 R:13 E:20 Retrieved:7 Score:0.913
403 R:13 E:24 Retrieved:7 Score:0.913
403 R:11 E:19 Retrieved:7 Score:0.908
403 R:12 E:21 Retrieved:7 Score:0.908
403 R:13 E:21 Retrieved:7 Score:0.908
403 R:13 E:23 Retrieved:7 Score:0.908
403 R:14 E:18 Retrieved:7 Score:0.908
403 R:13 E:18 Retrieved:7 Score:0.904
403 R:11 E:18 Retrieved:7 Score:0.903
```



```
405 R:5 E:24 Retrieved:3 Score:0.8055
405 R:5 E:25 Retrieved:3 Score:0.8055
405 R:5 E:3 Retrieved:3 Score:0.72222
405 R:5 E:4 Retrieved:3 Score:0.66666
405 R:5 E:19 Retrieved:3 Score:0.6324
405 R:5 E:21 Retrieved:3 Score:0.6324
405 R:5 E:22 Retrieved:3 Score:0.6269
405 R:5 E:23 Retrieved:3 Score:0.6269
405 R:5 E:20 Retrieved:3 Score:0.6222
405 R:6 E:19 Retrieved:3 Score:0.5833
```



```
408 R:1 E:20 Retrieved:1 Score:0.007936507936507936
408 R:5 E:1 Retrieved:1 Score:0.006944444444444444
408 R:6 E:1 Retrieved:1 Score:0.006944444444444444
408 R:1 E:1 Retrieved:1 Score:0.00641025641025641
408 R:2 E:1 Retrieved:1 Score:0.00641025641025641
408 R:17 E:1 Retrieved:1 Score:0.006172839506172839
408 R:1 E:2 Retrieved:0 Score:0.0
408 R:1 E:3 Retrieved:0 Score:0.0
408 R:1 E:4 Retrieved:0 Score:0.0
408 R:1 E:5 Retrieved:0 Score:0.0
```

Explanation:

Form our experiments, the settings lead to good performance for individual queries shows below:

- For query **401**: The value of R is 1,2,or from 11 to 17 and the value of E is less or equal than 3, then it gets the highest score.
- For query **402**: The value of R is between 9 to 16 and the value of E is 17,25,23,24,16, then it gets the highest score.
- For query **403**: The value of R is between 11 to 14 and the value of E is between 18 to 24, then it gets the highest score.
- For query **405**: The value of R is 5 and the value of E is 3,4,or between 19 to 25, then it gets the highest score.
- For query **408**: The value of R is 1,2,5,6,17 and the value of E is 1 or 20, then it gets the highest score.

From the statistic of these settings, it is hard to find the good settings lead to good performance on average. So I think the query expansion parameters R and E are dependent on different input query for these collection.

4.2 Improved Query Expansion

The ideas of improving query expansion:

- The most common way of query expansion is using some form of thesaurus. For each term in a query, the query can be automatically added some synonyms and related words for each term from some thesaurus dictionary resource. Using thesaurus can be combined with ideas of term weigh, but the weight of added terms should be less than original query terms.
- We can use a controlled vocabulary that is maintained by human editors. Using this vocabulary is quite common for well resourced domains, such as medical or legal domains.
- We can also use word co-occurrence statistics over a collection of documents in a domain which are used to automatically induce a thesaurus.
- In order to implement idea of thesaurus improvement, I have done some search work in this field. Here I have to mention WordNet project which is a large lexical database for English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. WordNet superficially resembles a thesaurus, in that it groups words together based on their meanings. So I think we can use this thesaurus as our expansion dictionary to improve input query.
- One more things I have to mention here is that because I have not added stemming function at indexing time. If we add stemming process into index program, I think it will improve the query performance. In this assignment, I have done the **porter stemmer function** in my **Stemmer.java** class which implements passing a sentence by calling stemmingOperation() function and it will return a stemmed sentence back.

Because I don't have enough time to implement all these functions, so I just give the ideas for this part. I may implement these ideas in the future to demonstrate the improvement for query expansion.

Reference:

1. Lecture Note.
2. Introduction to information retrieval, Cambridge Press 2008
3. Porter Stemmer from: www.tartarus.org/~martin/PorterStemmer/
4. Some code from my Assignment one.