1. Introduction

In this project, we have to implement several different retrieval methods. To finish this project, we need use the toolkit lemur or indri, even other useful toolkit. But lemur and indri are the most suitable to do this project. So, I choose **Indri-5.10** to finish this project. First, we must download the data collection called WT2g. This data collection contains 207491 different web documents, and have 2GB capacity.
I run this project in the OS-ubuntu-14.04, and to compile the Indri-5.10 correctly. First, I install the compile environment in the Ubuntu terminal. The following are commands that I used：

```
$ sudo apt-get install g++
$ sudo apt-get install zlib1g-dev
```

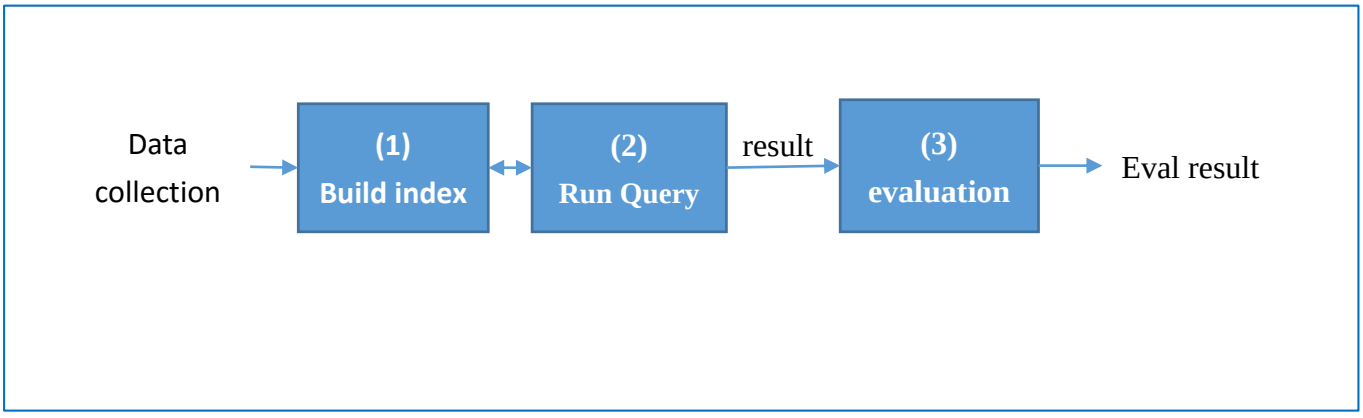
Second, I install Indri-5.10, the operation also in Ubuntu terminal. I used the following commands：

```
$ tar -zxf indri-5.10.tar.gz
$ cd indri-5.10
$ ./configure
$ make
$ sudo make install
```

After install Indri-5.10, we can start our task.
Basically, to complete this project, there have three steps：

(1) Build three index to our data collection.
  (a) without stemming
  (b) with porter stemming
  (c) with krovetz stemming
(2) Generate result file with four method, each query will return a ranked list of documents (the top 1000) in a particular format. So each file have at most 50000 result lines. Finally, we have 3*4=12 result files.
(3) Use **trec_eval** to evaluation each file and generate 12 evaluation results.

The following is flow diagram：



2. Indexing data collection

First, we have to build index to our data collection. In indri, it has a useful command called **IndriBuildIndex,** it can index document. This command is replace in the path：<Indri-5.10/buildindex/IndriBuildIndex>. The following is the command format：

$ IndriBuildIndex <parameter_file>

The content of the parameter file is shown on Figure 1, it has XML format.

```
parameter_file_porterstem.xml ×
<parameters>

    <index>/home/peter/WSM_Project2/index_porter_stemming</index>
    <memory>1G</memory>

    <stemmer>
        <name>porter</name>
    </stemmer>

    <corpus>
        <path>/home/peter/WSM_Project2/WT2G</path>
        <class>trecweb</class>
    </corpus>



    <field><name>p</name></field>

</parameters>
```

Figure 1

The **<index>** tags assign the construct index repository path, **<memory>** tags specifying the number of bytes to use for the indexing process. The **<stemmer>** tags specifying the stemming algorithm(Porter or Krovetz) while index collection. The **<corpus><path>**/path/to/file_or_directory**</path></corpus>** specifying the data collection path and **<corpus><class>trecweb</class></corpus>** specifying the file class environment is **trecweb.** Finally, in the project, I use the field **p** to build index.

I tried two stemming algorithm (porter and krovetz), the other is no stemming. So, I construct three different index. The result shown on Figure 2. And Figure 3 shown the index build process.

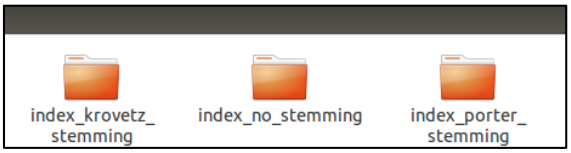
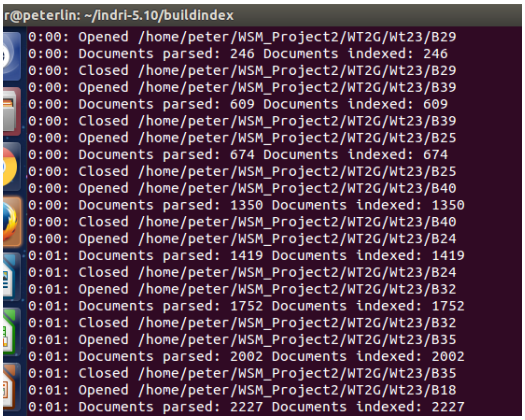


Figure 2



```
r@peterlin: ~/indri-5.10/buildindex
0:00: Opened /home/peter/WSM_Project2/WT2G/Wt23/B29
0:00: Documents parsed: 246 Documents indexed: 246
0:00: Closed /home/peter/WSM_Project2/WT2G/Wt23/B29
0:00: Opened /home/peter/WSM_Project2/WT2G/Wt23/B39
0:00: Documents parsed: 609 Documents indexed: 609
0:00: Closed /home/peter/WSM_Project2/WT2G/Wt23/B39
0:00: Opened /home/peter/WSM_Project2/WT2G/Wt23/B25
0:00: Documents parsed: 674 Documents indexed: 674
0:00: Closed /home/peter/WSM_Project2/WT2G/Wt23/B25
0:00: Opened /home/peter/WSM_Project2/WT2G/Wt23/B40
0:00: Documents parsed: 1350 Documents indexed: 1350
0:00: Closed /home/peter/WSM_Project2/WT2G/Wt23/B40
0:00: Opened /home/peter/WSM_Project2/WT2G/Wt23/B24
0:01: Documents parsed: 1419 Documents indexed: 1419
0:01: Closed /home/peter/WSM_Project2/WT2G/Wt23/B24
0:01: Opened /home/peter/WSM_Project2/WT2G/Wt23/B32
0:01: Documents parsed: 1752 Documents indexed: 1752
0:01: Closed /home/peter/WSM_Project2/WT2G/Wt23/B32
0:01: Opened /home/peter/WSM_Project2/WT2G/Wt23/B35
0:01: Documents parsed: 2002 Documents indexed: 2002
0:01: Closed /home/peter/WSM_Project2/WT2G/Wt23/B35
0:01: Opened /home/peter/WSM_Project2/WT2G/Wt23/B18
0:01: Documents parsed: 2227 Documents indexed: 2227
```

Figure 3

Then I want know the three indexes status, how many total terms and unique terms there have in each index? So I use the command dumpindex, the following is format：

$ dumpindex <index_path> s

Then the terminal shown each index status. See the Figure 4~6.

```
Repository statistics:
documents:        247491
unique terms:     1525012
total terms:      261397310
fields:           p
```

Figure 4：no stemming

```
Repository statistics:
documents:        247491
unique terms:     1341890
total terms:      261397310
fields:           p
```

Figure 5：porter stemming

```
Repository statistics:
documents:        247491
unique terms:     1392092
total terms:      261397310
fields:           p
```

Figure 6：krovetz stemming

3. Run Query and Evaluation

In the indri, it has a useful command called IndriRunQuery, the format is following：

$ IndriRunQuery [query_parameter_file] > result

The content of the query parameter file is shown on Figure 7, it has XML format.

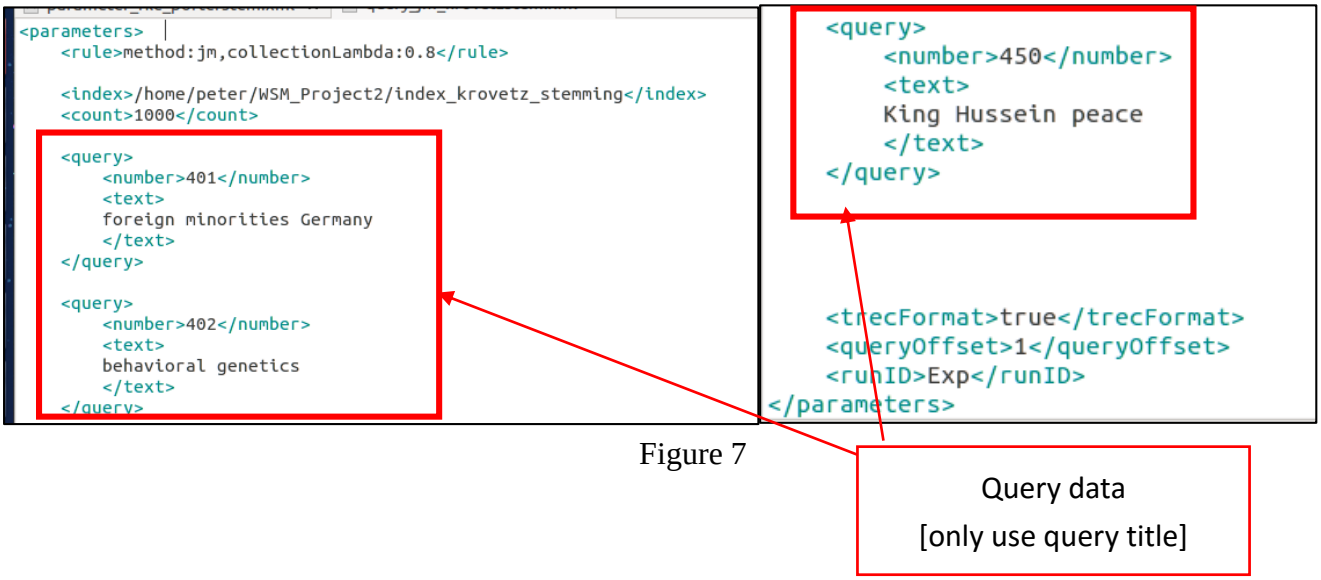


Figure 7

The <rule> tags specifying the smoothing method in language model, if we want to run query to other retrieval model, i.e. okapi(BM25) or tfidf. Then just change <rule> tags to <baseline> tags, the following is the baseline format.

<baseline>okapi,k1:1.0,b:0.3,k3:7</basesline>

The <trecFormat> tags specifying the output format is trecformat, the <queryOffset> tags specifying the runquery process starting query number, and the <runID> tags specifying a string specifying the id for a query run.

After use the command IndriRunQuery, I get 12 result files, that shown on Figure 8.

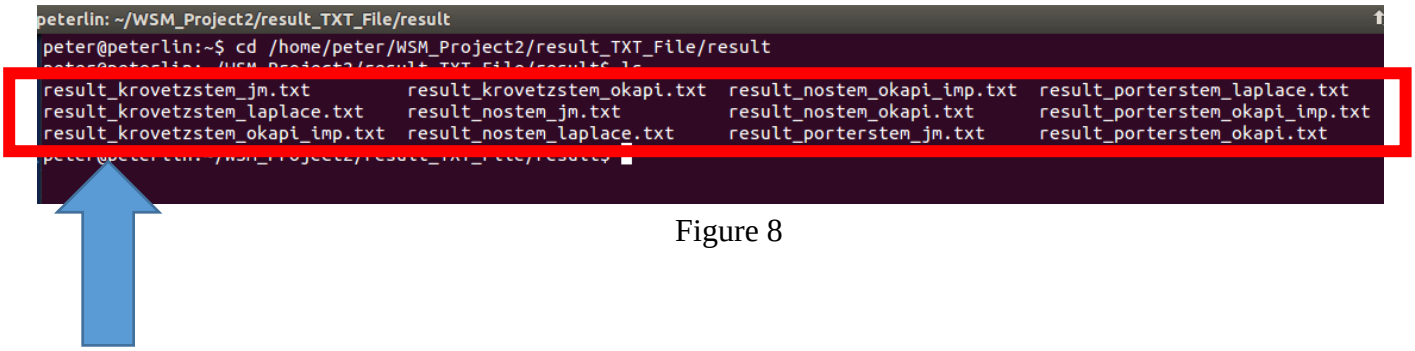


Figure 8

Each line in the result files has the following format, shown on Figure 9：

---

query-number Q0 document-id rank score Exp

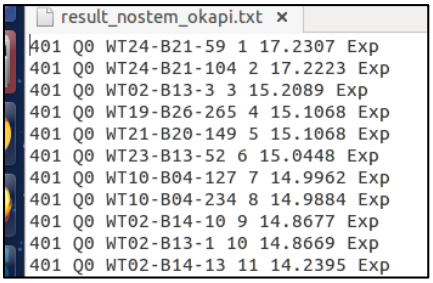


Figure 9

To evaluate the generate result, we need use the 12 result files in Figure 8. First, I download the **qrels.401-450** and **trec_eval** and then use the following command ：

Trec_eval –q qrels.401-450 trec_eval > eval_result

After use the command, I get 12 eval_result_files, that shown on Figure 10.

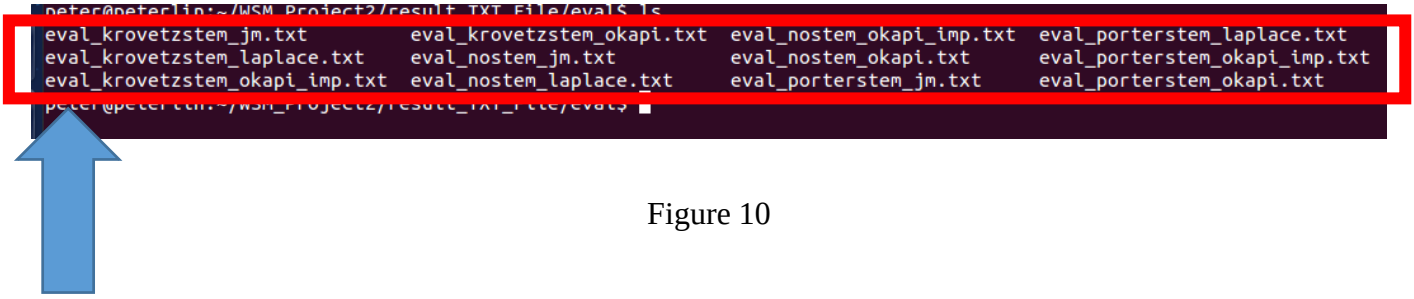


Figure 10

The following part 3.1~3.4, I descript the four method and how do I complete the task.

3.1. OKAPI TF-IDF

To do okapi TF-IDF, in each query file of okapi method must add following <baseline> tags

---

<baseline>okapi,k1:2.0,b:0.75,k3:0.0</baseline>

The okapi score formula is

$$RSV_d = \sum_{t \in q} \left[\log \frac{N}{\mathrm{df}_t} \right] \cdot \frac{(k_1+1)\mathrm{tf}_{td}}{k_1((1-b)+b \times (L_d/L_{ave}))+\mathrm{tf}_{td}} \cdot \frac{(k_3+1)\mathrm{tf}_{tq}}{k_3+\mathrm{tf}_{tq}}$$

It emphasizes the **inter-relationship between the query terms within a document.**

Because the task requirement, I set the k1 = 2.0, b = 0.75, but according above formula, we must ignore the fraction $\frac{(k_3+1)tf_{tq}}{k_3+tf_{tq}}$, so I set **k3 = 0.0**, than the value of this fraction is **1**

Then runquery and trec_eval will generate three result, shown on the Figure 11~13. I just show the evaluation result of all query, not of each query. The red square part is Un-interpolated mean average precision number and Precision at rank 10 documents.

```
Total number of documents over all queries
    Retrieved:    48182
    Relevant:      2279
    Rel_ret:       1429
Interpolated Recall - Precision Averages:
    at 0.00       0.7021
    at 0.10       0.5308
    at 0.20       0.4097
    at 0.30       0.3080
    at 0.40       0.2503
    at 0.50       0.1824
    at 0.60       0.1269
    at 0.70       0.0837
    at 0.80       0.0339
    at 0.90       0.0173
    at 1.00       0.0038
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.2152
Precision:
  At    5 docs:   0.4600
  At   10 docs:   0.3980
  At   15 docs:   0.3773
  At   20 docs:   0.3390
  At   30 docs:   0.2880
  At  100 docs:   0.1504
  At  200 docs:   0.0965
  At  500 docs:   0.0494
  At 1000 docs:   0.0286
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2750
```

Figure 11：no stemming

```
Total number of documents over all queries
    Retrieved:    48885
    Relevant:      2279
    Rel_ret:       1648
Interpolated Recall - Precision Averages:
    at 0.00       0.6685
    at 0.10       0.5331
    at 0.20       0.4266
    at 0.30       0.3362
    at 0.40       0.2776
    at 0.50       0.2275
    at 0.60       0.1670
    at 0.70       0.1093
    at 0.80       0.0451
    at 0.90       0.0137
    at 1.00       0.0046
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.2308
Precision:
  At    5 docs:   0.3800
  At   10 docs:   0.4040
  At   15 docs:   0.3720
  At   20 docs:   0.3380
  At   30 docs:   0.2953
  At  100 docs:   0.1688
  At  200 docs:   0.1125
  At  500 docs:   0.0576
  At 1000 docs:   0.0330
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2876
```

Figure 12：porter stemming

```
Total number of documents over all queries
    Retrieved:    48795
    Relevant:      2279
    Rel_ret:       1608
Interpolated Recall - Precision Averages:
    at 0.00       0.6792
    at 0.10       0.5446
    at 0.20       0.4472
    at 0.30       0.3449
    at 0.40       0.2935
    at 0.50       0.2324
    at 0.60       0.1707
    at 0.70       0.1111
    at 0.80       0.0467
    at 0.90       0.0170
    at 1.00       0.0070
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.2362
Precision:
  At    5 docs:   0.4120
  At   10 docs:   0.4020
  At   15 docs:   0.3680
  At   20 docs:   0.3380
  At   30 docs:   0.2927
  At  100 docs:   0.1690
  At  200 docs:   0.1112
  At  500 docs:   0.0568
  At 1000 docs:   0.0322
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2912
```

Figure 13：krovetz stemming

3.2. language model with Laplace smoothing

To do laplace smoothing, I modify the program code, adjusting the dirichlet smoothing formula. Following compare the change of the formula and code.

And use tags **<rule>method:dirichlet,mu: unique terms</rule>**, the unique terms is a number of each index contains unique terms.

The code path is **indri-5.10/include/indri/DirichletTermScoreFunction.hpp**

dirichlet smoothing	laplace smoothing
$P(r\|D) = \dfrac{tf_{r,D} + mu * P(r\|C)}{\|D\| + mu}$	$P(r\|D) = \dfrac{tf_{r,D} + 1}{\|D\| + mu}$

```
double scoreOccurrence( double occurrences, int contextSize ) {
  double seen = ( double(occurrences) + _muTimesCollectionFrequency ) / ( double(contextSize) + _mu );
  return log( seen );
}

double scoreOccurrence( double occurrences, int contextSize, double documentOccurrences, int documentLen
//two level Dir Smoothing!
//        tf_E + documentMu*P(t|D)
```

```
double scoreOccurrence( double occurrences, int contextSize ) {
  double seen = ( double(occurrences) + 1 ) / ( double(contextSize) + _mu );
  return seen;
}

double scoreOccurrence( double occurrences, int contextSize, double documentOcc
//two level Dir Smoothing!
//        tf_E + documentMu*P(t|D)
```

Then runquery and trec_eval will generate three result, shown on the Figure 14~16. I just show the evaluation result of all query, not of each query. The red square part is Un-interpolated mean average precision number and Precision at rank 10 documents.

```
Queryid (Num):       50
Total number of documents over all queries
    Retrieved:    48182
    Relevant:      2279
    Rel_ret:       1101
Interpolated Recall - Precision Averages:
    at 0.00       0.3257
    at 0.10       0.2086
    at 0.20       0.1329
    at 0.30       0.1095
    at 0.40       0.0897
    at 0.50       0.0635
    at 0.60       0.0493
    at 0.70       0.0346
    at 0.80       0.0187
    at 0.90       0.0078
    at 1.00       0.0014
Average precision (non-interpolated) for all rel docs(averaged over queries
                  0.0798

Precision:
  At    5 docs:   0.1280
  At   10 docs:   0.1160
  At   15 docs:   0.1147
  At   20 docs:   0.1120
  At   30 docs:   0.1087
  At  100 docs:   0.0716
  At  200 docs:   0.0538
  At  500 docs:   0.0332
  At 1000 docs:   0.0220
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.0973
```

Figure 14：no stemming

```
Queryid (Num):       50
Total number of documents over all queries
    Retrieved:    48885
    Relevant:      2279
    Rel_ret:       1173
Interpolated Recall - Precision Averages:
    at 0.00       0.2574
    at 0.10       0.1756
    at 0.20       0.1191
    at 0.30       0.0959
    at 0.40       0.0810
    at 0.50       0.0618
    at 0.60       0.0463
    at 0.70       0.0338
    at 0.80       0.0200
    at 0.90       0.0109
    at 1.00       0.0027
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.0699
Precision:
  At    5 docs:   0.1000
  At   10 docs:   0.1120
  At   15 docs:   0.1080
  At   20 docs:   0.1050
  At   30 docs:   0.0973
  At  100 docs:   0.0668
  At  200 docs:   0.0516
  At  500 docs:   0.0352
  At 1000 docs:   0.0235
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.0919
```

Figure 15：porter stemming

```
Total number of documents over all queries
    Retrieved:    48795
    Relevant:      2279
    Rel_ret:       1210
Interpolated Recall - Precision Averages:
    at 0.00       0.2852
    at 0.10       0.1962
    at 0.20       0.1317
    at 0.30       0.1121
    at 0.40       0.0869
    at 0.50       0.0666
    at 0.60       0.0499
    at 0.70       0.0339
    at 0.80       0.0209
    at 0.90       0.0116
    at 1.00       0.0028
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.0782
Precision:
  At    5 docs:   0.1200
  At   10 docs:   0.1200
  At   15 docs:   0.1147
  At   20 docs:   0.1140
  At   30 docs:   0.1040
  At  100 docs:   0.0744
  At  200 docs:   0.0556
  At  500 docs:   0.0372
  At 1000 docs:   0.0242
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.1021
```

Figure 16：krovetz stemming

3.3. Language modeling with Jelinek-Mercer smoothing

This method involves a linear interpolation of the maximum likelihood model with the collection model, using a coefficient lambda to control the influence.

The formula is **(1-lambda)*P_{ml}(w|d)+lambda*P(w|c)**

I use the tags **<rule>method:jm,collectionLambda:0.8</rule>**, just set lambda value = 0.8.

Then runquery and trec_eval will generate three result, shown on the Figure 17~19. I just show the evaluation result of all query, not of each query. The red square part is Un-interpolated mean average precision number and Precision at rank 10 documents.

```
Total number of documents over all queries
    Retrieved:    48182
    Relevant:      2279
    Rel_ret:       1440
Interpolated Recall - Precision Averages:
    at 0.00       0.5911
    at 0.10       0.4202
    at 0.20       0.3270
    at 0.30       0.2766
    at 0.40       0.2162
    at 0.50       0.1666
    at 0.60       0.1278
    at 0.70       0.0974
    at 0.80       0.0657
    at 0.90       0.0508
    at 1.00       0.0301
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.1905
Precision:
  At    5 docs:   0.3120
  At   10 docs:   0.3040
  At   15 docs:   0.2960
  At   20 docs:   0.2770
  At   30 docs:   0.2467
  At  100 docs:   0.1430
  At  200 docs:   0.0938
  At  500 docs:   0.0496
  At 1000 docs:   0.0288
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2441
```

Figure 17：no stemming

```
Total number of documents over all queries
    Retrieved:    48885
    Relevant:      2279
    Rel_ret:       1583
Interpolated Recall - Precision Averages:
    at 0.00       0.5604
    at 0.10       0.4211
    at 0.20       0.3303
    at 0.30       0.2665
    at 0.40       0.2160
    at 0.50       0.1827
    at 0.60       0.1434
    at 0.70       0.1123
    at 0.80       0.0730
    at 0.90       0.0496
    at 1.00       0.0279
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.1922
Precision:
  At    5 docs:   0.3040
  At   10 docs:   0.2900
  At   15 docs:   0.2720
  At   20 docs:   0.2570
  At   30 docs:   0.2307
  At  100 docs:   0.1462
  At  200 docs:   0.1004
  At  500 docs:   0.0544
  At 1000 docs:   0.0317
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2220
```

Figure 18：porter stemming

```
Total number of documents over all queries
    Retrieved:    48795
    Relevant:      2279
    Rel_ret:       1586
Interpolated Recall - Precision Averages:
    at 0.00       0.5723
    at 0.10       0.4424
    at 0.20       0.3456
    at 0.30       0.2793
    at 0.40       0.2217
    at 0.50       0.1846
    at 0.60       0.1471
    at 0.70       0.1170
    at 0.80       0.0738
    at 0.90       0.0488
    at 1.00       0.0290
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.1987
Precision:
  At    5 docs:   0.3200
  At   10 docs:   0.3040
  At   15 docs:   0.2787
  At   20 docs:   0.2660
  At   30 docs:   0.2393
  At  100 docs:   0.1472
  At  200 docs:   0.1028
  At  500 docs:   0.0536
  At 1000 docs:   0.0317
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2377
```

Figure 19：krovetz stemming

3.4. improve one of the above three IR models

I chose improve the first method OKAPI TF-IDF. In the first method. In the first method, we must ignore the fraction $\frac{(k_3+1)tf_{tq}}{k_3+tf_{tq}}$, so set **k3 = 0.0**, now let us consider fraction $\frac{(k_3+1)tf_{tq}}{k_3+tf_{tq}}$, that is say don't set the value of k3=0.0. I find a web page(http://nlp.stanford.edu/IR-book/html/htmledition/okapi-bm25-a-non-binary-model-1.html). In this page, it mentioned what is the reasonable values. So I extract some contents below：

"In the absence of such optimization, experiments have shown reasonable values are to set k1 and k3 to a value between 1.2 and 2 and b = 0.75."

So I try to set <baseline> tags below：

<baseline>okapi,k1:1.2,b:0.75,k3:1.2</baseline>

Then runquery and trec_eval will generate three result, shown on the Figure 20~22. I just show the evaluation result of all query, not of each query. The red square part is Un-interpolated mean average precision number and Precision at rank 10 documents.

Finally compare the result values of Average precision and R-Precision with the first method, it can find the two values are higher than first method. So set this group of parameters can improve the retrieve effects.

```
Total number of documents over all queries
    Retrieved:    48182
    Relevant:      2279
    Rel_ret:       1498
Interpolated Recall - Precision Averages:
    at 0.00       0.6960
    at 0.10       0.5506
    at 0.20       0.4352
    at 0.30       0.3320
    at 0.40       0.2762
    at 0.50       0.1972
    at 0.60       0.1569
    at 0.70       0.1129
    at 0.80       0.0540
    at 0.90       0.0355
    at 1.00       0.0140
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.2362

Precision:
  At    5 docs:   0.4560
  At   10 docs:   0.4300
  At   15 docs:   0.3920
  At   20 docs:   0.3430
  At   30 docs:   0.3027
  At  100 docs:   0.1628
  At  200 docs:   0.1022
  At  500 docs:   0.0527
  At 1000 docs:   0.0300
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2916
```

Figure 20：no stemming

```
Total number of documents over all queries
    Retrieved:    48885
    Relevant:      2279
    Rel_ret:       1745
Interpolated Recall - Precision Averages:
    at 0.00       0.6635
    at 0.10       0.5439
    at 0.20       0.4490
    at 0.30       0.3531
    at 0.40       0.2987
    at 0.50       0.2497
    at 0.60       0.1968
    at 0.70       0.1472
    at 0.80       0.0859
    at 0.90       0.0316
    at 1.00       0.0151
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.2498

Precision:
  At    5 docs:   0.3960
  At   10 docs:   0.4100
  At   15 docs:   0.3720
  At   20 docs:   0.3470
  At   30 docs:   0.3000
  At  100 docs:   0.1786
  At  200 docs:   0.1193
  At  500 docs:   0.0608
  At 1000 docs:   0.0349
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.3038
```

Figure 21：porter stemming

```
Total number of documents over all queries
    Retrieved:    48795
    Relevant:      2279
    Rel_ret:       1702
Interpolated Recall - Precision Averages:
    at 0.00       0.6677
    at 0.10       0.5544
    at 0.20       0.4614
    at 0.30       0.3550
    at 0.40       0.3055
    at 0.50       0.2528
    at 0.60       0.1936
    at 0.70       0.1467
    at 0.80       0.0776
    at 0.90       0.0319
    at 1.00       0.0144
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.2530

Precision:
  At    5 docs:   0.4200
  At   10 docs:   0.4080
  At   15 docs:   0.3667
  At   20 docs:   0.3440
  At   30 docs:   0.2987
  At  100 docs:   0.1798
  At  200 docs:   0.1172
  At  500 docs:   0.0589
  At 1000 docs:   0.0340
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.3048
```

Figure 22：krovetz stemming

3.5. Conclusion

According above results, only consider the values of Average precision and R-Precision, it can find stemming while constructing indexes can get better results than without stemming usually. This is advantage of stemming. But maybe sometimes stemming will remove more information leading the worst results. It is disadvantage.

Next, we compare results of Laplace smoothing and Jelinek-Mercer smoothing. It can find phenomenon that Jelinek-Mercer smoothing seem to get better retrieve effects than Laplace smoothing in this data collection. Because it Average precision and R-Precision are higher than Laplace smoothing.

4. Extra runs

I also try different values of lambda with Jelinek-Mercer smoothing. Following list the result of two different lambda values(**lambda = 0.0, lambda = 0.4**), below shown six result of the three indexes(Figure 23~28).

```
Total number of documents over all queries
    Retrieved:    48182
    Relevant:      2279
    Rel_ret:       1222
Interpolated Recall - Precision Averages:
    at 0.00       0.5781
    at 0.10       0.3997
    at 0.20       0.3247
    at 0.30       0.2897
    at 0.40       0.2236
    at 0.50       0.1560
    at 0.60       0.1384
    at 0.70       0.0993
    at 0.80       0.0624
    at 0.90       0.0378
    at 1.00       0.0254
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.1882
Precision:
  At    5 docs:   0.3640
  At   10 docs:   0.3300
  At   15 docs:   0.3040
  At   20 docs:   0.2720
  At   30 docs:   0.2407
  At  100 docs:   0.1374
  At  200 docs:   0.0854
  At  500 docs:   0.0430
  At 1000 docs:   0.0244
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2379
```

Figure 23：no stemming(**lambda = 0.0**)

```
Total number of documents over all queries
    Retrieved:    48885
    Relevant:      2279
    Rel_ret:       1484
Interpolated Recall - Precision Averages:
    at 0.00       0.6111
    at 0.10       0.4850
    at 0.20       0.3830
    at 0.30       0.3167
    at 0.40       0.2654
    at 0.50       0.2113
    at 0.60       0.1587
    at 0.70       0.1273
    at 0.80       0.0958
    at 0.90       0.0429
    at 1.00       0.0278
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.2219
Precision:
  At    5 docs:   0.3720
  At   10 docs:   0.3440
  At   15 docs:   0.3173
  At   20 docs:   0.2910
  At   30 docs:   0.2573
  At  100 docs:   0.1626
  At  200 docs:   0.1038
  At  500 docs:   0.0546
  At 1000 docs:   0.0297
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2668
```

Figure 24：porter stemming(**lambda = 0.0**)

```
Total number of documents over all queries
    Retrieved:    48795
    Relevant:      2279
    Rel_ret:       1458
Interpolated Recall - Precision Averages:
    at 0.00       0.6137
    at 0.10       0.4907
    at 0.20       0.4063
    at 0.30       0.3137
    at 0.40       0.2715
    at 0.50       0.2159
    at 0.60       0.1582
    at 0.70       0.1259
    at 0.80       0.0931
    at 0.90       0.0407
    at 1.00       0.0288
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.2230
Precision:
  At    5 docs:   0.3880
  At   10 docs:   0.3460
  At   15 docs:   0.3147
  At   20 docs:   0.2830
  At   30 docs:   0.2587
  At  100 docs:   0.1600
  At  200 docs:   0.1022
  At  500 docs:   0.0524
  At 1000 docs:   0.0292
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2750
```

Figure 25：krovetz stemming(**lambda = 0.0**)

```
Total number of documents over all queries
    Retrieved:    48182
    Relevant:      2279
    Rel_ret:       1544
Interpolated Recall - Precision Averages:
    at 0.00       0.6150
    at 0.10       0.4664
    at 0.20       0.3856
    at 0.30       0.3247
    at 0.40       0.2683
    at 0.50       0.2081
    at 0.60       0.1645
    at 0.70       0.1216
    at 0.80       0.0888
    at 0.90       0.0687
    at 1.00       0.0416
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.2253
Precision:
  At    5 docs:   0.3720
  At   10 docs:   0.3460
  At   15 docs:   0.3307
  At   20 docs:   0.3050
  At   30 docs:   0.2700
  At  100 docs:   0.1684
  At  200 docs:   0.1059
  At  500 docs:   0.0547
  At 1000 docs:   0.0309
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2727
```

Figure 26：no stemming(**lambda = 0.4**)

```
Total number of documents over all queries
    Retrieved:    48885
    Relevant:      2279
    Rel_ret:       1729
Interpolated Recall - Precision Averages:
    at 0.00       0.5839
    at 0.10       0.4715
    at 0.20       0.3922
    at 0.30       0.3198
    at 0.40       0.2688
    at 0.50       0.2311
    at 0.60       0.1806
    at 0.70       0.1470
    at 0.80       0.1047
    at 0.90       0.0675
    at 1.00       0.0372
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.2302
Precision:
  At    5 docs:   0.3320
  At   10 docs:   0.3400
  At   15 docs:   0.3213
  At   20 docs:   0.3030
  At   30 docs:   0.2560
  At  100 docs:   0.1680
  At  200 docs:   0.1165
  At  500 docs:   0.0612
  At 1000 docs:   0.0346
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2687
```

Figure 27：porter stemming(**lambda = 0.4**)

```
Total number of documents over all queries
    Retrieved:    48795
    Relevant:      2279
    Rel_ret:       1719
Interpolated Recall - Precision Averages:
    at 0.00       0.5889
    at 0.10       0.4800
    at 0.20       0.4000
    at 0.30       0.3274
    at 0.40       0.2706
    at 0.50       0.2233
    at 0.60       0.1856
    at 0.70       0.1503
    at 0.80       0.1054
    at 0.90       0.0692
    at 1.00       0.0380
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.2332
Precision:
  At    5 docs:   0.3640
  At   10 docs:   0.3460
  At   15 docs:   0.3280
  At   20 docs:   0.2960
  At   30 docs:   0.2547
  At  100 docs:   0.1708
  At  200 docs:   0.1158
  At  500 docs:   0.0594
  At 1000 docs:   0.0344
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2694
```

Figure 28：krovetz stemming(**lambda = 0.4**)

Compare with the third method, we can found the phenomenon that along with the lambda value decrease, the Average precision and R-Precision are increase obviously.

Finally, I also run the twostage smoothing method, so I set the <rule> tags below：

<rule>method:twostage,mu: unique terms,lambda:0.5</rule>

The twostage smoothing is 2-component mixture, it mixture the Dirichlet smoothing and Laplace smoothing. The value of mu, I set the number of unique terms in the three indexes and set lambda = 0.5.

Then I get three results shown on Figure 29~31.

```
Total number of documents over all queries
    Retrieved:    48182
    Relevant:      2279
    Rel_ret:       1605
Interpolated Recall - Precision Averages:
    at 0.00       0.6037
    at 0.10       0.4220
    at 0.20       0.3389
    at 0.30       0.2836
    at 0.40       0.2408
    at 0.50       0.2043
    at 0.60       0.1584
    at 0.70       0.1338
    at 0.80       0.1074
    at 0.90       0.0789
    at 1.00       0.0521
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.2107
Precision:
  At    5 docs:   0.3280
  At   10 docs:   0.3020
  At   15 docs:   0.2813
  At   20 docs:   0.2720
  At   30 docs:   0.2427
  At  100 docs:   0.1530
  At  200 docs:   0.1043
  At  500 docs:   0.0558
  At 1000 docs:   0.0321
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2379
```

Figure 29：no stemming

```
Total number of documents over all queries
    Retrieved:    48885
    Relevant:      2279
    Rel_ret:       1808
Interpolated Recall - Precision Averages:
    at 0.00       0.5279
    at 0.10       0.4023
    at 0.20       0.3240
    at 0.30       0.2699
    at 0.40       0.2221
    at 0.50       0.1972
    at 0.60       0.1561
    at 0.70       0.1337
    at 0.80       0.1109
    at 0.90       0.0865
    at 1.00       0.0519
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.2011
Precision:
  At    5 docs:   0.3120
  At   10 docs:   0.2800
  At   15 docs:   0.2693
  At   20 docs:   0.2640
  At   30 docs:   0.2400
  At  100 docs:   0.1524
  At  200 docs:   0.1116
  At  500 docs:   0.0630
  At 1000 docs:   0.0362
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2306
```

Figure 30：porter stemmings

```
Total number of documents over all queries
    Retrieved:    48795
    Relevant:      2279
    Rel_ret:       1787
Interpolated Recall - Precision Averages:
    at 0.00       0.5380
    at 0.10       0.4118
    at 0.20       0.3285
    at 0.30       0.2867
    at 0.40       0.2375
    at 0.50       0.2024
    at 0.60       0.1548
    at 0.70       0.1349
    at 0.80       0.1137
    at 0.90       0.0880
    at 1.00       0.0534
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.2070
Precision:
  At    5 docs:   0.3160
  At   10 docs:   0.2860
  At   15 docs:   0.2773
  At   20 docs:   0.2560
  At   30 docs:   0.2420
  At  100 docs:   0.1510
  At  200 docs:   0.1115
  At  500 docs:   0.0626
  At 1000 docs:   0.0357
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2369
```

Figure 31：krovetz stemming

Compare with the Laplace smoothing and Jelinek-Mercer smoothing in the above four method, we can found a phenomenon. The twostage smoothing result values of Average precision and R-Precision are better than Laplace smoothing and approach to the Jelinek-Mercer smoothing. So set mu = unique terms and lambda:0.5 with the twostage smoothing, it result is not bad.

5. Recall/precision graphs

In this part, I will show the Groups column chart of the average precision of the above four methods. Following shown on Figure 32~35.

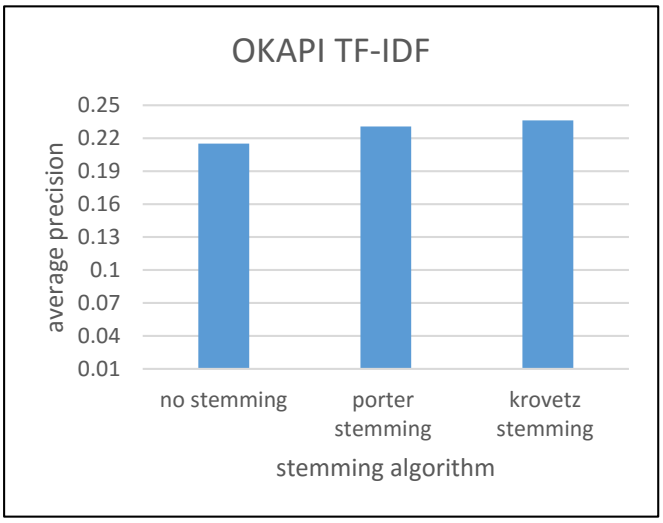


Figure 32

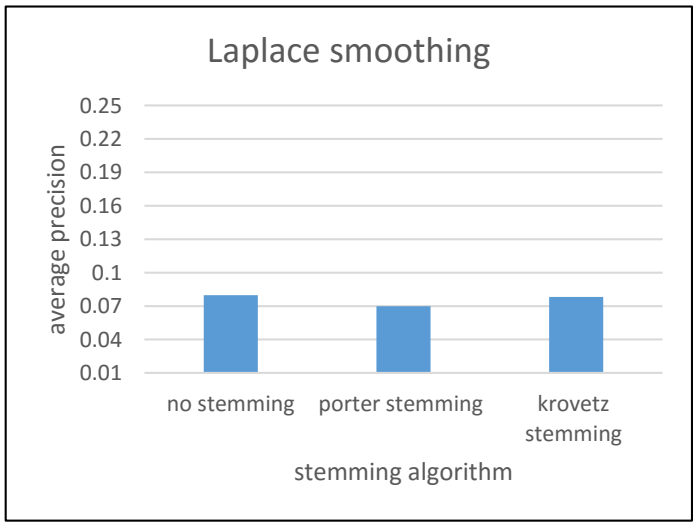


Figure 33

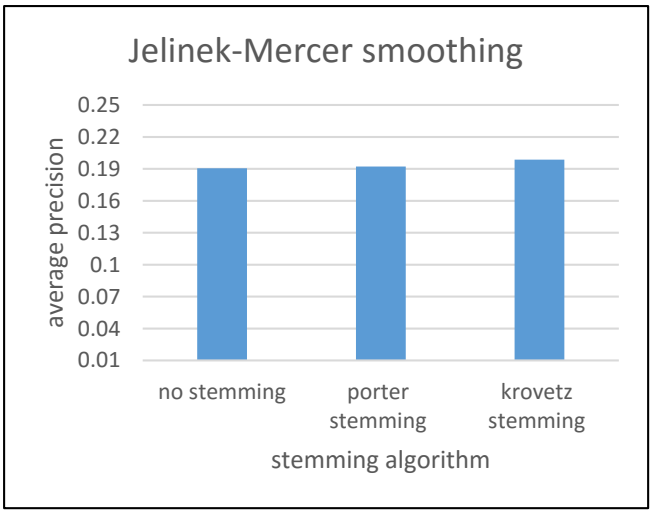


Figure 34

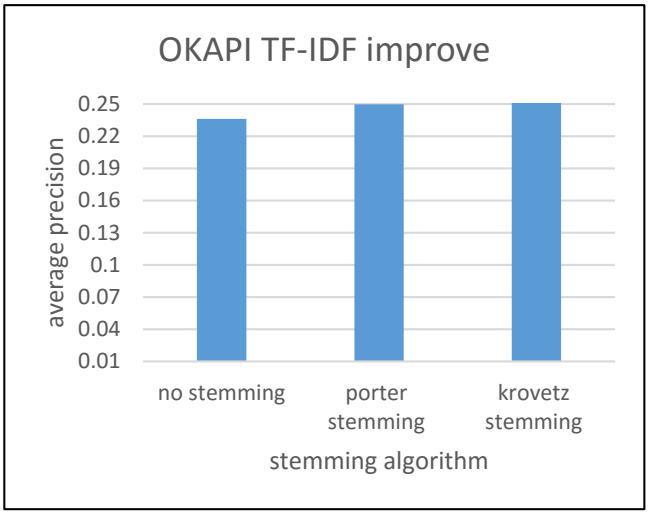


Figure 35

There is show clearly. The laplace smoothing average precision lower than other methods obviously. In the fourth method, the average precision are higher than first method. It improve retrieve effect obviously.

Following, I show the Interpolated Recall – Precision graphs of above four method with no stemming and porter stemming. Shown on Figure 36~43.

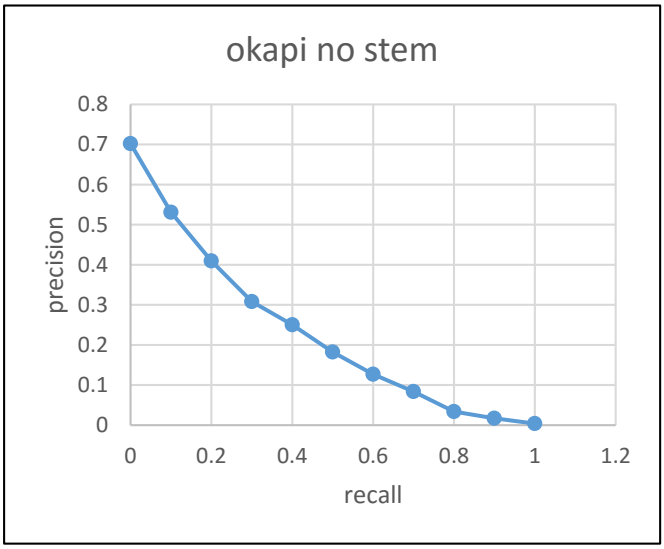


Figure 36

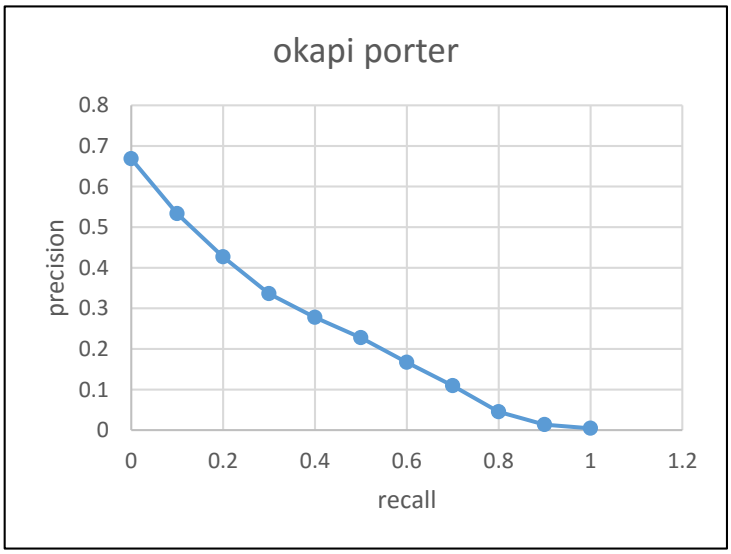


Figure 37

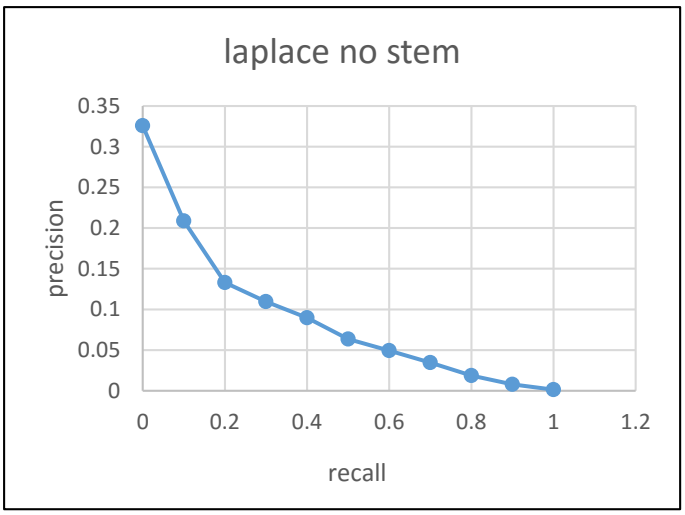


Figure 38

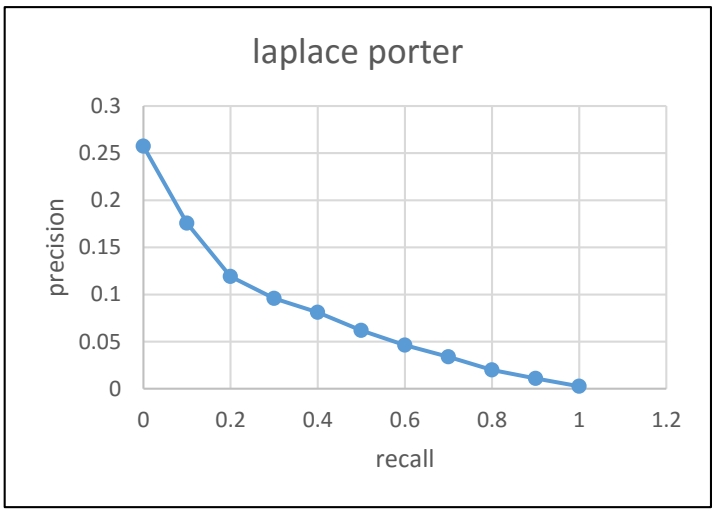


Figure 39

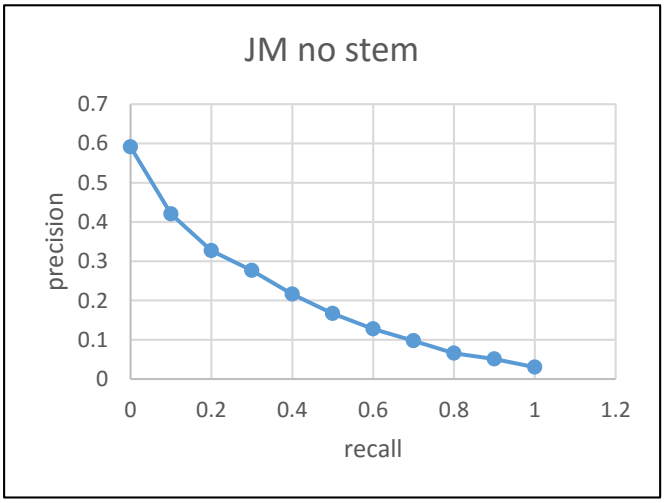


Figure 40

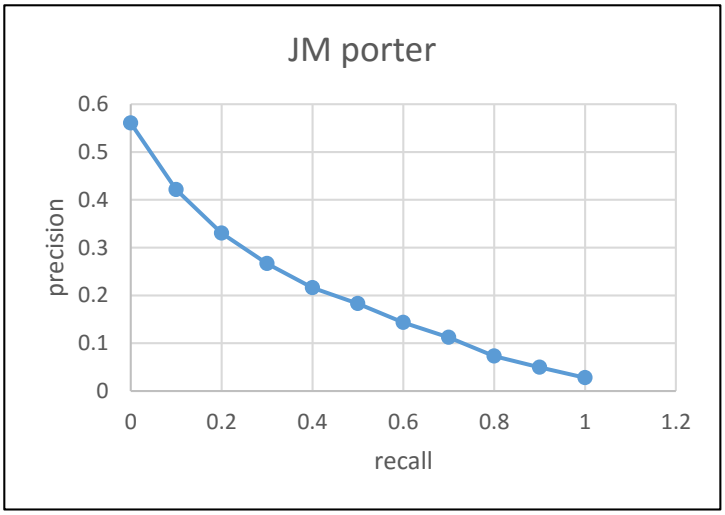


Figure 41

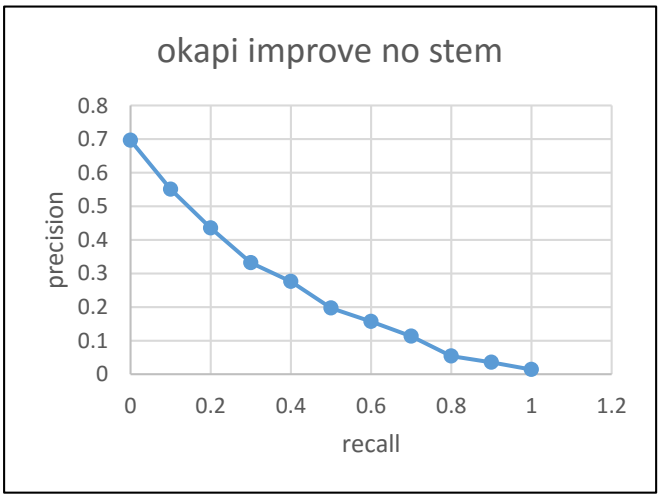


Figure 42

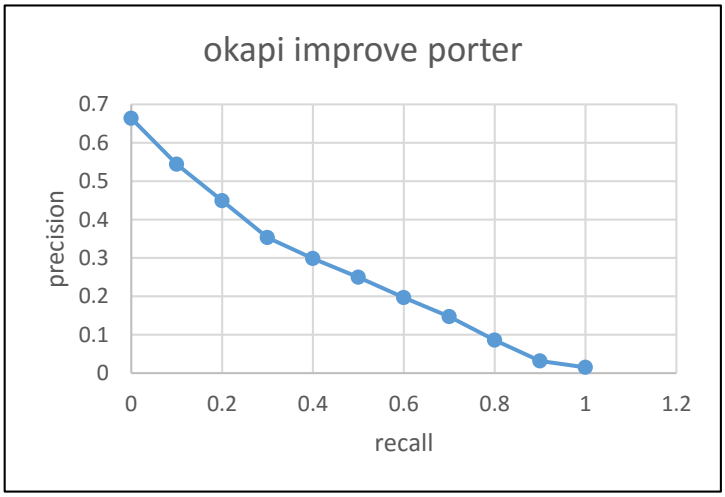
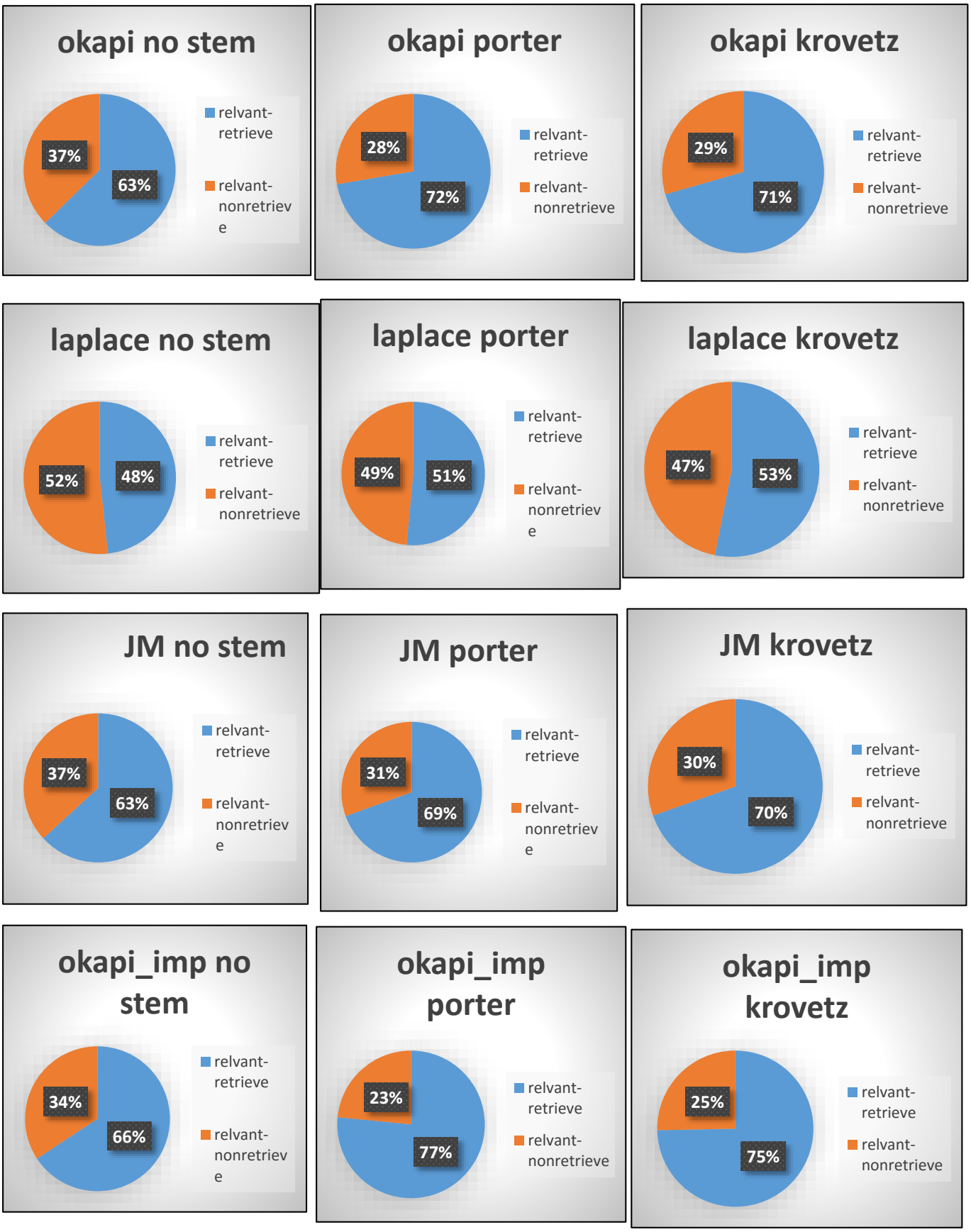


Figure 43

Then the above eight result graphs it make sense, along with the recall value increasing, the precision value is decreasing.

6. Failure Analysis

Following I show 12 result Pie chart of percent of relvant-retrieve and relvant-nonretrieve.



Compare with stemming (porter or krovetz) and without stemming. The with stemming results, it percent of relvant-retrieve are higher than without stemming. So with stemming can improve the result effects.