Your Name: Hailun Zhu

Your Andrew ID: hailunz

Homework 4

Collaboration and Originality

1. Did you receive help of any kind from anyone in developing your software for this assignment (Yes or No)? It is not necessary to describe discussions with the instructor or TAs.
   No.
   If you answered Yes, provide the name(s) of anyone who provided help, and describe the type of help that you received.

2. Did you give help of any kind to anyone in developing their software for this assignment (Yes or No)?
   No.
   If you answered Yes, provide the name(s) of anyone that you helped, and describe the type of help that you provided.

3. Are you the author of every line of source code submitted for this assignment (Yes or No)? It is not necessary to mention software provided by the instructor.
   Yes.
   If you answered No:
   a. identify the software that you did not write,
   b. explain where it came from, and
   c. explain why you used it.

4. Are you the author of every word of your report (Yes or No)?
   Yes.
   If you answered No:
   a. identify the text that you did not write,
   b. explain where it came from, and
   c. explain why you used it.

Your Name: Hailun Zhu

Your Andrew ID: hailunz

Homework 4

1 Experiment 1: Baselines

Provide information about the effectiveness of your system in five baseline configurations.

	Ranked Boolean AND	Indri			
		BOW		Query Expansion	
		Your System	Reference System	Your System	Reference System
P@10	0.3250	0.3200	0.3100	0.3000	0.2950
P@20	0.3675	0.3600	0.3525	0.3200	0.3275
P@30	0.3700	0.3467	0.3467	0.3467	0.3433
MAP	0.1881	0.1865	0.1883	0.1886	0.1908
win/loss	N/A	15/5	15/5	13/7	13/7

Document the parameter settings that were used to obtain these results.

Indri:mu=500
Indri:lambda=0.1
fb=true
fbDocs=10
fbTerms=10
fbMu=0
fbOrigWeight=0.5
fbExpansionQueryFile=HW4-train-1.qryOut
fbInitialRankingFile=Indri-Bow.teIn

Comment on the quality and character of the query expansion terms that were included, and the weights that were produced. Do they seem reasonable? Provide information about a few example queries to make your points, for example queries that had the most dramatic change in performance (good or bad) from query expansion, but do not provide information about every query individually. We are primarily interested in your observations about general trends, not quirky queries.

The original terms in the origin query usually are included in the expended query, and they have highest weights among all the expended terms. And the other expended query terms are usually content related terms or synonyms. It usually is reasonable. The documents used to retrieve these terms must have high scores by evaluating the original query, which implies they might have more terms in the original queries. And the documents with high scores are usually more related to the query, so that their terms may more related to the original query terms.

For example, for the 37 query- pampered chef. The top 2 terms in the expended query with highest weights are pamper and chef, which refer to the original query. And the other terms include recipe, bake, kitchen, which are topic related terms with chef. For some queries the expansion works well, those expanded query terms are more related to the original query term, this is the case with "cheap internet" and "avp", etc. The query "avp" gains the most improvement. This maybe because the original query is too short to have enough information and the expanded terms make a complement to this. But for query like "Obama family tree", the expanded query make performance worse. The key idea maybe the "family tree" and "obama", but the expanded terms turned out to be just partially related to the topic, like surname and history, so that the improper expansion may make the result worse.

Example queries:

1 : #wand (0.1194 family 0.0856 tree 0.0210 genealogy 0.0168 surname 0.0165 trivia 0.0157 crest 0.0129 engrave 0.0124 history 0.0113 byte 0.0097 barack)

37 : #wand (0.2585 chef 0.1102 pamper 0.0209 stoneware 0.0189 personal 0.0189 recipe 0.0153 stone 0.0151 bake 0.0127 pan 0.0123 pamperedchef 0.0122 kitchen)

52 : #wand (0.8824 avp 0.1506 newsletter 0.0934 tour 0.0695 antiviru 0.0550 croc 0.0515 event 0.0510 2008 0.0477 2007 0.0442 onsite 0.0432 season)

Comment on the effects of query expansion on your system and on the reference system. Are the two systems affected equally by query expansion, or are there important differences?

The effects of query expansion on my system and the reference system are almost the same. The map value has increased due to query expansion, but the P@n values are all slightly decreased, this may be because when we add the expanded query, we expand the search range so that we could retrieve more relevant documents, thus improve the recall but hurt the precision a little bit. And the win/loss ratios also decreased a little. And the query expansion works in a same way to these queries- the result of "Obama family tree" decreased most, and that of the "avp" improved most.

2 Experiment 2: The number of feedback documents

Provide information about the effect of the number of feedback documents on query expansion.

	Ranked Boolean AND	**Indri BOW, Your System**	**Query Expansion, Your Initial Results**					
			Feedback Documents					
			10	**20**	**30**	**40**	**50**	**100**
P@10	0.3250	0.3200	0.3000	0.2950	0.2850	0.2950	0.2900	0.2850
P@20	0.3675	0.3600	0.3200	0.3025	0.3125	0.3075	0.3175	0.3175
P@30	0.3700	0.3467	0.3467	0.3383	0.3317	0.3517	0.3500	0.3400
MAP	0.1881	0.1865	0.1886	0.1897	0.1897	0.1924	0.1925	0.1857
win/loss	N/A	15/5	13/7	15/5	13/7	14/6	15/5	13/7

	Ranked Boolean AND	Indri BOW, Reference System	Query Expansion, Reference System Initial Results					
			Feedback Documents					
			10	20	30	40	50	100
P@10	0.3250	0.3100	0.2950	0.3100	0.2800	0.3050	0.3050	0.3000
P@20	0.3675	0.3525	0.3275	0.3350	0.3100	0.3225	0.3250	0.3300
P@30	0.3700	0.3467	0.3433	0.3517	0.3450	0.3483	0.3500	0.3333
MAP	0.1881	0.1883	0.1908	0.1924	0.1865	0.1900	0.1913	0.1868
win/loss	N/A	15/5	13/7	13/7	15/5	15/5	14/6	13/7

Document the values of any parameters that were held constant during this experiment.

Indri:mu=500
Indri:lambda=0.1
fb=true
fbDocs=X
fbTerms=10
fbMu=0
fbOrigWeight=0.5
fbExpansionQueryFile=HW4-train-1.qryOut
fbInitialRankingFile=Indri-Bow.teIn

Comment on the effect of varying the number of feedback documents on the quality and character of the query expansion terms that were included, and the weights that were produced. Were any values consistently better than other values? Does using more documents tend to help the results, or hurt the results? Why? Provide information about a few example queries to make your points, for example queries that had the most dramatic change in performance as the number of documents varied, but do not provide information about every query individually. We are primarily interested in your observations about general trends, not quirky queries. If using more documents improves expansion quality, is the improvement worth the added computational costs?

Increasing the number of feedback documents will increase the weights of the terms, especially for terms that have highest weights before. And these one or two terms will rarely change due to the increasing number of feedback documents. The top terms generally will still be the terms that in the original query. They are stable to some degree. But changing the number of feedback documents will affect the following terms, especially for those ranked low in the expanded query and have small weights. Their weights are closer to each other, so that changing the document number could affect their orders and then the expanded query could be slightly different. For some queries that gain improvement, their expansion queries have better quality terms than before.

For example, for query 52, which gains the most dramatic improvement:

Number:10

52 : #wand (0.8824 avp 0.1506 newsletter 0.0934 tour 0.0695 antiviru 0.0550 croc 0.0515 event 0.0510 2008 0.0477 2007 0.0442 onsite 0.0432 season)

Number:50

52 : #wand (1.4036 avp 0.1586 newsletter 0.1182 tour 0.1026 croc 0.0901 alien 0.0750 predator 0.0712 volleyball 0.0695 antiviru 0.0693 beach 0.0691 rfc3588)

Number:100

52 : #wand (1.6951 avp 0.1634 newsletter 0.1362 tour 0.1354 croc 0.1212 alien 0.0969 predator 0.0922 volleyball 0.0860 planet 0.0855 2008 0.0803 beach)

The first three terms are the same among these tests. Increasing number of documents will increase all the weights. The last few terms or orders of the terms are different.

For query 1, whose performance decreased most:

Number 10:

1 : #wand (0.1194 family 0.0856 tree 0.0210 genealogy 0.0168 surname 0.0165 trivia 0.0157 crest 0.0129 engrave 0.0124 history 0.0113 byte 0.0097 barack)

Number: 50

1 : #wand (1.0813 family 0.3682 tree 0.3446 surname 0.3434 crest 0.3414 genealogy 0.2504 history 0.0343 6 0.0323 5 0.0291 3 0.0270 4)

Number 100:

1 : #wand (1.2991 family 0.4422 tree 0.4069 genealogy 0.4047 surname 0.4035 crest 0.3012 history 0.1227 obama 0.0607 barack 0.0388 5 0.0386 6)

For the result of my system, having 50 feedback documents gets the best MAP value as well as win/loss ratio. For the reference system, having 20 feedback documents gets the best MAP value and having 40 feedback documents gets the best win/loss ratios, but if taking these two measurement into account, having 50 feedback documents could also be considered to be a best setting to balance the MAP and win/loss ratio. When increasing document number from 10 to 50, by adding more documents, we may consider more relevant documents to get the related terms, so that the expanded queries are more useful to complement to the original ones. As number is 50, it gains a high P@n(when n=30) and MAP value, this indicates we have a high recall performance. But when we increased number to 100, the performance dramatically declined. This may be because we are taking more than enough documents into account. The documents ranked low from the first evaluation may not be so related to the query, and so that their terms may be improper to describe the original query. The redundant terms expand the search range and influence the term scores so that hurt the overall performance.

By increasing the number of the feedback documents, we need to compute more term scores in the new added documents, thus the computational costs will increase. For number of 10 to 50, the improvement worth the added computational costs.

Comment on the effects of query expansion on your system and on the reference system. Are the two systems affected equally by query expansion, or are there important differences?

The general effects of varying the number of feedback documents are similar. For the expanded queries, increasing the documents number increases the terms' weights. The first two or three terms and their orders are often stable. The following terms and the orders are quite likely to be changed. And when using 100 feedback documents the performance decrease a lot.

3 Experiment 3: The number of feedback terms

Provide information about the effect of the number of feedback terms on query expansion.

	Ranked Boolean AND	Indri BOW, Your System	Query Expansion, Your Initial Results					
			Feedback Terms					
			5	10	20	30	40	50
P@10	0.3250	0.3200	0.2850	0.2900	0.3200	0.3200	0.3250	0.3200
P@20	0.3675	0.3600	0.3125	0.3175	0.3525	0.3600	0.3550	0.3625
P@30	0.3700	0.3467	0.3383	0.3500	0.3550	0.3650	0.3667	0.3667
MAP	0.1881	0.1865	0.1791	0.1925	0.2006	0.2006	0.2008	0.2022
Win/loss	N/A	15/5	14/6	15/5	15/5	15/5	15/5	15/5

	Ranked Boolean AND	Indri BOW, Reference System	Query Expansion, Reference System Initial Results					
			Feedback Terms					
			5	10	20	30	40	50
P@10	0.3250	0.3100	0.2800	0.3050	0.3000	0.3050	0.3050	0.3050
P@20	0.3675	0.3525	0.3000	0.3250	0.3475	0.3475	0.3500	0.3550
P@30	0.3700	0.3467	0.3350	0.3500	0.3550	0.3700	0.3650	0.3667
MAP	0.1881	0.1883	0.1738	0.1913	0.1941	0.1983	0.1990	0.1986
Win/loss	N/A	15/5	14/6	14/6	15/5	15/5	15/5	15/5

Document the values of any parameters that were held constant during this experiment.

Indri:mu=500
Indri:lambda=0.1
fb=true
fbDocs=50
fbTerms=X
fbMu=0
fbOrigWeight=0.5
fbExpansionQueryFile=HW4-train-1.qryOut
fbInitialRankingFile=Indri-Bow.teIn

Comment on the effect of varying the number of feedback terms on the quality and character of the query expansion terms that were included, and the weights that were produced. Were any values consistently better than other values? Does using more terms tend to help the results, or hurt the results? Why? Provide information about a few example queries to make your points, for example queries that had the

most dramatic change in performance as the number of terms varied. If using more terms improves expansion quality, is the improvement worth the added computational costs?

The quality and character of the query expansion terms are similar to the previous experiment, because we are just including more terms to the query. The top terms in the expanded query are usually terms in the original query, and their weights are often much higher than the others. This is reasonable, the documents used to retrieve these terms must have high score by evaluating the original query, which implies they must have more terms in the original queries.

By increasing the number of terms used, the query that has the most dramatic improvement in my system changed to be the No 26 query.

Terms 10:

26 : #wand (0.1462 heart 0.1396 rate 0.0369 lower 0.0366 your 0.0357 monitor 0.0259 exercise 0.0225 pulse 0.0187 rest 0.0185 target 0.0181 zone)

Terms 30:

26 : #wand (0.1462 heart 0.1396 rate 0.0369 lower 0.0366 your 0.0357 monitor 0.0259 exercise 0.0225 pulse 0.0187 rest 0.0185 target 0.0181 zone 0.0173 you 0.0164 beat 0.0154 training 0.0149 age 0.0147 maximum 0.0142 fitness 0.0131 polar 0.0122 hrmax 0.0117 minute 0.0117 use 0.0108 disease 0.0107 bpm 0.0105 can 0.0102 calculate 0.0096 mhr 0.0094 reebok 0.0091 intensity 0.0088 sensational 0.0082 measure 0.0079 per)

Terms 50:

26 : #wand (0.1462 heart 0.1396 rate 0.0369 lower 0.0366 your 0.0357 monitor 0.0259 exercise 0.0225 pulse 0.0187 rest 0.0185 target 0.0181 zone 0.0173 you 0.0164 beat 0.0154 training 0.0149 age 0.0147 maximum 0.0142 fitness 0.0131 polar 0.0122 hrmax 0.0117 minute 0.0117 use 0.0108 disease 0.0107 bpm 0.0105 can 0.0102 calculate 0.0096 mhr 0.0094 reebok 0.0091 intensity 0.0088 sensational 0.0082 measure 0.0079 per 0.0079 60 0.0078 blood 0.0078 from 0.0076 cardiosport 0.0076 cardiovascular 0.0075 number 0.0071 during 0.0070 pressure 0.0068 have 0.0068 one 0.0066 athlete 0.0064 about 0.0064 hotel 0.0064 method 0.0063 re 0.0062 artery 0.0061 health 0.0061 formula 0.0060 upper 0.0060 watch)

In my system, increasing the number of feedback terms, the P@n, and MAP are both increasing. For query 26, the extra terms that are added to the expanded query are very related to the original query terms, such as beat, health, disease, etc. As the “lower heart rate” may be health related topic, and are very likely to appear together with phrases like “age”, “heart beat”, so that including more those terms could improve the retrieval performance.

The 1st query’s performance falls most. But among those experiments, the decline in MAP compared to the RankedBoolean one is the same. Adding more terms does not make it better. This may be because those terms are nonrelated to the query terms, thus there is no improvement on the retrieval performance.

1 : #wand (1.0813 family 0.3682 tree 0.3446 surname 0.3434 crest 0.3414 genealogy 0.2504 history 0.0343 6 0.0323 5 0.0291 3 0.0270 4 0.0227 2 0.0198 7 0.0165 trivia 0.0130 8 0.0129 engrave 0.0113

byte 0.0103 gift 0.0097 barack 0.0090 obama 0.0087 9 0.0069 jordan 0.0069 your 0.0068 00 0.0067 his 0.0065 10 0.0061 birthstone 0.0061 michael 0.0060 record 0.0053 345 0.0051 ancestor 0.0048 free 0.0048 linkpendium 0.0046 rank 0.0046 biography 0.0045 11 0.0043 12 0.0042 politics 0.0040 29 0.0040 about 0.0039 oct 0.0038 knowledge 0.0037 name 0.0035 07 0.0035 test 0.0035 take 0.0034 tag 0.0033 13 0.0033 2008 0.0032 am 0.0032 dukes)

In my system, using more terms generally could improve the precision and recall performance. I have also tested number with 60, the map and win/loss ratio still increases. I think the improvement worth the added computational costs.

Comment on the effects of query expansion on your system and on the reference system. Are the two systems affected equally by query expansion, or are there important differences?

From number 5-40 the performance trend of the reference system is similar with my system's, however, there is a big difference, that is when number is 50, the performance declined in reference system, whereas improved in my system. Then I separately tested with parameter set to 60. The result shows that the performance of my system still improved whereas the reference system's still declined. I think this is related to the quality of the retrieved documents of the original query and the parameter settings for the Indri. When the fbdocs=50, my system got the best performance, but the reference system doesn't. As for the reference system, the best term number seems to be 40. For reference system, including more terms may have negative effect because the added terms are not closely related to the query. It is reasonable to have a "peak" value. As increasing the number of terms in my system, I think there should also be a threshold beyond which the performance will decrease.

4 Experiment 4: Original query vs. expanded query

Provide information about the effect of varying the weight between the original query and the new expansion query.

	Ranked Boolean AND	**Indri BOW, Your System**	**Query Expansion, Your Initial Results**					
			fbOrigWeight					
			0.0	**0.2**	**0.4**	**0.6**	**0.8**	**1.0**
P@10	0.3250	0.3200	0.3150	0.3200	0.3200	0.3250	0.3200	0.3200
P@20	0.3675	0.3600	0.3475	0.3575	0.3525	0.3525	0.3700	0.3600
P@30	0.3700	0.3467	0.3600	0.3617	0.3617	0.3667	0.3600	0.3467
MAP	0.1881	0.1865	0.1945	0.2035	0.2045	0.1978	0.1927	0.1865
Win/loss	N/A	15/5	13/7	15/5	15/5	16/4	15/5	15/5

	Ranked Boolean AND	Indri BOW, Reference System	Query Expansion, Reference System Initial Results					
			fbOrigWeight					
			0.0	0.2	0.4	0.6	0.8	1.0
P@10	0.3250	0.3100	0.2950	0.3000	0.3100	0.3300	0.3200	0.3100
P@20	0.3675	0.3525	0.3400	0.3525	0.3475	0.3525	0.3625	0.3525
P@30	0.3700	0.3467	0.3633	0.3550	0.3667	0.3633	0.3567	0.3467
MAP	0.1881	0.1883	0.1919	0.2000	0.2010	0.1958	0.1917	0.1883
Win/loss	N/A	15/5	14/6	14/6	15/5	15/5	15/5	15/5

Document the values of any parameters that were held constant during this experiment.

Indri:mu=500
Indri:lambda=0.1
fb=true
fbDocs=50
fbTerms=40
fbMu=0
fbOrigWeight=X
fbExpansionQueryFile=HW4-train-1.qryOut
fbInitialRankingFile=Indri-Bow.teIn

Comment also on the balance between the original query and the expansion query. Is a combination of the two queries worthwhile? Why or why not? How does the stability (win/loss) behavior compare to just using the expanded query alone?

Combination of the original query and the expansion query could get improvement both on recall and precision value. For both of my system and reference system, 0.4 is the best weight to get the best MAP value. By adding the expansion query, we are adding more information to the query thus to expand the range of searching and matching, so that the recall value are increasing by increasing the weights of the expanded query. However, there are terms in the expansion query that are not so proper to describe the original query, thus only using expansion query won't get the best performance. We still need the original query to give the original terms, that is the core terms higher weights in order to distinguish them from the expansion terms. But only using original query is also too strict to get a good performance.

By combining the two queries, the stability (win/loss) ratio is higher. And increasing the original query weight the win/loss ratio remains stable. This may be because the quality of expansion queries for different queries are varied. So that using the expanded query alone will show this difference. Raising the original query weight could reduce this effect. The original query is more stable than the expanded query, because there is no biased term, all the terms in the original query are important terms.

Comment on the effects of the combined query on your system and on the reference system. Are the two systems affected equally, or are there important differences?

The effects of the combined query on my system and on the reference system are almost the same. They are affected equally. For both these systems, weight 0.4 gets the best performance. Decreasing or increasing the original weights from 0.4 both undermined the MAP value.

5 Experiment 5: Effect of the original query quality

Provide information about the effect of varying the weight between the original query and the new expansion query

	Ranked Boolean AND	Query Expansion, Your Initial Results			
		BOW Original Query		SDM Original Query	
		Original	Expanded	Original	Expanded
P@10	0.3250	0.3200	0.3200	0.4000	0.4950
P@20	0.3675	0.3600	0.3525	0.4425	0.4900
P@30	0.3700	0.3467	0.3617	0.4617	0.4983
MAP	0.1881	0.1865	0.2045	0.2459	0.2899
Win/loss	N/A	15/5	15/5	14/6	15/5

	Ranked Boolean AND	Query Expansion, Reference System Initial Results			
		BOW Original Query		SDM Original Query	
		Original	Expanded	Original	Expanded
P@10	0.3250	0.3100	0.3100	0.4000	0.4650
P@20	0.3675	0.3525	0.3475	0.4200	0.4925
P@30	0.3700	0.3467	0.3667	0.4400	0.4867
MAP	0.1881	0.1883	0.2010	0.2231	0.2815
Win/loss	N/A	15/5	15/5	15/5	16/4

Document the values of the parameters used for this experiment.

Indri:mu=500
Indri:lambda=0.1
fb=true
fbDocs=50
fbTerms=40
fbMu=0
fbOrigWeight=0.4
fbExpansionQueryFile=HW4-train-1.qryOut
fbInitialRankingFile=Indri-Bow.teIn

Does a difference in the quality of the initial retrieval make any difference in query expansion effectiveness or stability?

Yes, there is a difference, the better the initial retrieval is, the better effect the query expansion will get. For both my system and the reference system, the performances of the query expansion are the best among all the experiments. The P@n, MAP and win/loss ratio are all the highest. This means that the precision, recall and stability are all improved. This shows that the quality of the original query really matters to the final retrieval performance. More accurate and proper query could get a more reasonable initial retrieval, in which documents are more relevant to the information need whose contents are more

likely to be related to it. Thus, taking top ranked terms from these documents could form an expansion query which describes the information need in a more proper way.

Also, the improvement between the query expansion and the original query are larger in SDM than in original query. This shows that the effectiveness and stability of the query expansion will be improved by using a higher quality original query.

6 Analysis of results

You ran a lot of experiments, and have a lot of experimental results. The sections above discuss each experiment individually. In this section, we want you to think about general trends that you observed across the 5 experiments that have not been discussed in earlier sections.

How did query expansion affect the "high Precision" portion of a document ranking (the top-ranked documents) and the "high Recall" portion of the document ranking (farther down the ranking)? Where does query expansion have the greatest impact?

Was query expansion stable in your experiments (as indicated by the win/loss ratio)? Were any experimental conditions more or less stable? Was there a correlation between accuracy metrics and stability?

Is the increased computational complexity worth the increased accuracy (if any)? Keep in mind that a "production" implementation of pseudo relevance feedback would be much more optimized and faster than your implementation.

Feel free to include other comments about what you observed. You did a lot of experiments. This is your opportunity to let us know what you learned in this assignment.

From the previous experiments, the results shows that the query expansion will affect the "high Precision" portion of a document ranking more than the "high Recall" portion of the document ranking. The P@n(n=10,20,30) will fluctuate more along with the different performance of the query expansion, whereas the P@n(n=200,500,1000) will remain almost the same, or they will just changed a little bit.

This maybe because that by changing parameters, like fbdocs, fbterms, fbOriweights, the expanded queries could be slightly different, thus, the orders of some documents are different, which makes the P@n fluctuate when n is small. This will affect the high precision portion. However, the overall recall performance maybe more related to the original query, in other words, there is a maximum number of relevant documents that we could retrieve based on the original query, and as the expanded query are formed based on the initial retrieval, the major constraint is the original query. Through all these experiments, I find out that for experiment 1 to 4, the P@1000 value are very similar, which is around 0.342, meanwhile the P@1000 value for SDM expanded query are much higher, which is around 0.380. The huge performance in P@1000 may be produced by the better quality query.

The query expansion has the greatest impact on high Precision portion of a document ranking.

In experiment 2, the query expansion is less stable. I find out that the win/loss ratio changes a lot by changing the number of the feedback documents. In experiment 3 and 4, the query expansion is almost stable.

In experiment 2, increasing the fbdocs sometimes will increase the P@n and MAP, sometimes will decrease them, and this is the same for the trend of the win/loss ratio. When the accuracy metrics are not stable, the query expansion is not stable either. This may be because that that the query expansion on these queries has very different effects. It improves some queries but impairs the others, which may lead to the fluctuates in P@n value and the win/loss ratio.

I think the increased computational complexity worth the increased accuracy. For all the experiments, we could find a best setting to gain improvement. Considering the dramatic improvement that experiment 5 yields, I think it is worthwhile to implement the pseudo relevance feedback.

Through all these 5 experiments, I think the most important thing we should consider when using pseudo relevance feedback is to decide how to set those parameters. As can be seen from those experiment data, if we improperly set the parameters, the retrieval performance could be undermined. Using too much or too few feedback documents, feedback terms or weights won't get the best result. We need to find a balance between the stability, precision and recall. What's more, the query expansion's performance depends on the original query's quality. This is also a crucial factor that influence the query expansion effects.