Your Name: Jiangtian Qian

Your Andrew ID: jiangtiq

Homework 3

Collaboration and Originality

1. Did you receive help <u>of any kind</u> 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 <u>of any kind</u> 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 <u>every line</u> 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 <u>every word</u> 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: Jiangtian Qian

Your Andrew ID: jiangtiq

Homework 3

Instructions

1 Experiment 1: Baselines

	Ranked Boolean AND	Indri			
		BOW		Query Expansion	
		Your System	Reference System	Your System	Reference System
P@10	0.3789	0.5684	0.6000	0.6316	0.5947
P@20	0.3763	0.5368	0.5447	0.5711	0.5605
P@30	0.3632	0.4807	0.5228	0.5123	0.5211
MAP	0.0891	0.1363	0.1483	0.1677	0.1628
win/loss	N/A	15/4	15/4	16/3	16/3

1.1 Parameters

RankedBoolean AND	trecEvalOutputLength=100
My system of BOW	trecEvalOutputLength=100 Indri:mu=1000 Indri:lambda=0.7
Reference of BOW	Nothing to set

My system of query Expansion	trecEvalOutputLength=100
	Indri:mu=1000
	Indri:lambda=0.7
	fb=true
	fbDocs=10
	fbTerms=10
	fbMu=0
	fbOrigWeight=0.5

Reference System of query expansion	trecEvalOutputLength=100
	Indri:mu=1000
	Indri:lambda=0.7
	fb=true
	fbDocs=10
	fbTerms=10
	fbMu=0
	fbOrigWeight=0.5
	fbInitialRankingFile=Indri-Bow.teIn

The parameter setting are showing above. The fb, fbDocs, fbTerms and fbMu is setting as the handout write. And I set mu to 1000 and lambda to 0.7 as baseline for Indri. Because large lambda means less smoothing and better for the short query and all of our queries are short, so I set lambda to 0.7 for baseline. I set mu to 1000 because it gives a normal performance for Indri model, not too high MAP&P@n and not too low MAP&P@n, it is convenient for us to compare it with results under different mu later. Also, the large mu is more important to short documents and our retrieval documents are mostly long documents, so I set it to a small number, like 1000.

1.2 Discussion

As we can see, RankedBoolean AND get the worst performance compare to the other four models. It is quiet make sense. RankedBoolean AND is an exact match model, document is included in answer if each terms found in the document, it will cause a very low recall for we may missing relevant documents just because it doesn't have exact same terms. And documents sorted order is related to the frequency of the terms in the document, this cause a low precision in the top N for we may have a lot of term Q which is not that determined(important) in the query terms and this will increase the document score.

And both BOW model of Indri have high precision and MAP. This is reasonable. Indri is best match model, which has better performance of precision and MAP than exact match, because we will retrieval most of relevant documents in best match model.

And the query expansion model has the highest precision of top N and MAP as we can see. We use machine learning tools to create new query to retrieval model. The new query combine with the old one in a #wand(weight original (1 - weight) expansion query) model. Because we choose the top N terms in the top-ranked documents to create expansion query, it is more possible to have query terms included in the relevant documents. And then we combine it with the original query, which make sure we will retrieval right documents. So, both the precision and recall will be improved by using machine learning methods. Though some of RankedBoolean may have good MAP but the average of it is worse than Indri. Though the MAP or the average effective is increased, some queries are harmed, like the loss query.

For the win/loss proportion, BOW of Indri has 15/4 for both. Though we will improve our precision and recall with the Indri model, some query like "723: Executive privilege", "727: Church arson", "738: Anthrax hoaxes" and "741: Artificial Intelligence" has lower MAP than RankedBoolean. As we can see, these four query have high possibility to be phrase or terminology, only documents which has each terms can express the complete meaning of query and can be defined relevant. When we use Indri, other than exact match, the documents may just contain one of terms in the query, for example, documents contain "hoaxes" only for query "Anthrax hoaxes" many times, which can get high score but non-relevant. And when it comes to exact match, only documents include both words of phrases and terminology can have high score. The query expansion of two system has both win/loss 16/3. The loss query is "727: Church arson", "738: Anthrax hoaxes" and "741: Artificial Intelligence" and they loss for same reason we mentioned above. Though they loss, they are almost near to RankedBoolean. We can see machine learning methods truly improve a lot in precision and MAP.

At last, the difference between my system and reference system is quite little, seems like my system works well.

2 Experiment 2: The number of feedback documents

	Ranked Boolean AND	Indri BOW, Reference System	Query Expansion, Reference System Initial Results					
			Feedback Documents					
			10	20	30	40	50	100
P@10	0.3789	0.6000	0.5947	0.5895	0.6000	0.6000	0.6000	0.6158
P@20	0.3763	0.5447	0.5605	0.5579	0.5342	0.5605	0.5632	0.5684
P@30	0.3632	0.5228	0.5211	0.5246	0.5123	0.5298	0.5386	0.5263
MAP	0.0891	0.1483	0.1628	0.1598	0.1569	0.1608	0.1603	0.1581
win/loss	N/A	15/4	16/3	16/3	16/3	16/3	16/3	16/3

2.1 Parameters

RankedBoolean AND	trecEvalOutputLength=100
Reference of BOW	Nothing to set

Reference System of query expansion	trecEvalOutputLength=100
	Indri:mu=1000
	Indri:lambda=0.7
	fb=true
	fbDocs=10,20,30,40,50,100
	fbTerms=10
	fbMu=0
	fbOrigWeight=0.5
	fbInitialRankingFile=Indri-Bow.teIn

The parameter setting are showing above. The fb, fbTerms and fbMu is setting as the handout write. And I set mu to 1000 and lambda to 0.7 as baseline for Indri. Because large lambda means less smoothing and better for the short query and all of our queries are short, so I set lambda to 0.7 for baseline. I set mu to 1000 because it gives a normal performance for Indri model, not too high MAP&P@n and not too low MAP&P@n, it is convenient for us to compare it with results under different mu later. And set these parameter as same as baseline is convenient to compare the result. Also, the large mu is more important to short documents and our retrieval documents are mostly long documents, so I set it to a small number, like 1000. And for fbDocs, I set it from 10 to 100 as the table shows.

2.2 Discussion

As far as we know there is no good theory about how many documents to use for pseudo relevance feedback. The most common solution is the top N documents and N is based on the guess of the quality of the initial retrieval results, like the number of documents and the quality of the documents. Obviously, as we can see from the result table, the precision and MAP improve as feedback documents increase. This is reasonable. If we have more judged documents, we will have a larger dataset of candidate terms. And that gives more terms a chance to be top N. So, it is more stable and reliable for a system to get top N candidate terms. In this case, we will not miss too much truly relevant terms and include non relevant terms. Like query "gas supply cold region", top 10 documents may give high score to terms like "ice", "snow". When it comes to top 100 judged documents, the terms are more comprehensive and reasonable. Though the precision increase, it not increases too much, which means our original query's quality and top N judged documents' quality is good. And we can see from the query text, most of our query terms are noun, which is good for information need expression.

Though the precision improved, the MAP decreased, that because of the decrease of recall. When we increase the dataset of candidates terms, the score calculation related to tf * idf, it will compensate the frequency in the documents and the high score of the top 10 which may because of the high frequency now will get a lower score. And high frequency terms in top 10, top 20 may means they exists in lots of documents but now have low score and get low weight. So we will retrieval less relevant documents with expansion query now, which means lower recall.

As for win/loss, we have same win/loss for query expansion. It means judged documents number has little influence of big change in MAP. However, they are very close to the baseline. We should treat fbDocs as a collection-independent parameter to be tuned. And we can see the system is quiet stable when the fbDocs changes.

And the win/loss is better win/loss than Indri BOW and RankedBoolean. It is reasonable. The loss query is "727: Church arson", "738: Anthrax hoaxes" and "741: Artificial Intelligence". As we can see, these three loss query have high possibility to be phrase or terminology, only documents which has each terms can express the complete meaning of query and can be defined relevant. When we use Indri, other than exact match, the documents may just contain one of terms in the query, for example, documents contain "hoaxes" only for query "Anthrax hoaxes" many times, which can get high score but non-relevant. And when it comes to exact match, only documents include both words of phrases and terminology can have high score. The Indri's query has less recall than query expansion system, so the win/loss is worse. But this system need more time to run according to the increase of the judged documents, there is trade of accuracy and time efficiency. At last, we can see when fbDocs is 40, we get best performance.

3 Experiment 3: The number of feedback terms

	Ranked Boolean AND	**Indri BOW, Reference System**	**Query Expansion, Reference System Initial Results**					
			Feedback Terms					
			5	**10**	**20**	**30**	**40**	**50**
P@10	0.3789	0.6000	0.5895	0.5947	0.6053	0.6105	0.6000	0.6053
P@20	0.3763	0.5447	0.5211	0.5605	0.5500	0.5605	0.5526	0.5553
P@30	0.3632	0.5228	0.4842	0.5211	0.5281	0.5298	0.5228	0.5193
MAP	0.0891	0.1483	0.1437	0.1628	0.1635	0.1633	0.1627	0.1612
Win/loss	N/A	15/4	15/4	16/3	16/3	16/3	16/3	16/3

3.1 Parameters

The parameter setting are showing above. The fb, fbDocs and fbMu is setting as the handout write. And I set mu to 1000 and lambda to 0.7 as baseline for Indri. Because large lambda means less smoothing and better for the short query and all of our queries are short, so I set lambda to 0.7 for baseline. I set mu to 1000 because it gives a normal performance for Indri model, not too high MAP&P@n and not too low MAP&P@n, it is convenient for us to compare it with results under different mu later. And for fbTerms, I set it from 5 to 50 as the table shows.

RankedBoolean AND	trecEvalOutputLength=100
Reference of BOW	Nothing to set

Reference System of query expansion	trecEvalOutputLength=100
	Indri:mu=1000
	Indri:lambda=0.7
	fb=true
	fbDocs=10
	fbTerms=5, 10, 20, 30, 40, 50
	fbMu=0
	fbOrigWeight=0.5
	fbInitialRankingFile=Indri-Bow.teIn

3.2 Discussion

The suitable number of terms use for the expansion terms is related to the number of documents used for the query terms. If there are lots of documents, there are more evidence for selecting terms and a larger candidate vocabulary. As we can see from the table, when feedback terms number is 5, we get a worse performance than other, even Indri BOW. I get a look of the expansion query when feedback terms is 5, it gives a lot of weight to meaningless query terms like “we” in query 705 and its MAP is much lower than other expansion query under different feedback terms. The same as query 733, it gives too much weight to terms like 0.0016 to “your” and 0.0020 to “you”, which contributes nothing to the relevant results. These two just two examples and there are lots of similar situations in feedback terms equals 5 model. This means, two less feedback terms will harm the query results for it may include un-relevant query terms and directed to totally different results. So we have low precision, low recall and low WIN/LOSS in this situation.

The best number of terms various by query. Picking the right number yields significant improvements. As the fbTerms number increase, we get the best performance in 20 and 30. It improves a lot than fbTerms 5. When I take a lot at the expansion query under fbTerms 20, I do find some un-relevant query terms in it, but its weight is small for we have 20 query terms (each term will be assign less weight when we have same total weight for expansion query). So the un-relevant weight will have less effect in the final results. And when we have more terms, we will have higher proportion of relevant terms, which will help increase the precision and recall. So, the precision of top N and MAP both increase. When the fbTerms increase to 40 and 50, the precision and MAP decrease. That’s because, the original terms are short, when we expanded the query to 40 or 50 long, we could have lots of query terms un-relevant to information need. This will decrease both precision and recall. And as we can see, the fbTerms between 20 and 30 have similar performance and fbTerms under 20 will use less time to compute, which is a better choice.

The WIN/LOSS is quiet stable after fbTerms 10, that means average MAP is quiet stable after fbTerms 10. The most unstable queries are "727: Church arson" and "721: Census data applications". They change a lot under different fbTerms. This may because these two's information need are too ambiguous, or may because of every terms in query has weak connection.

And it is really time consuming to deal with more than 20 expansion query terms, I used 752178.782ms to get the results when fbTerms equals 40 and 897453.303ms to get results for 50 expansion query terms.

4 Experiment 4: Original query vs. expanded query

	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	0.3789	0.6000	0.5632	0.5947	0.5947	0.5842	0.6000	0.5684
P@20	0.3763	0.5447	0.5105	0.5316	0.5526	0.5605	0.5447	0.5368
P@30	0.3632	0.5228	0.4947	0.5175	0.5175	0.5175	0.5018	0.4807
MAP	0.0891	0.1483	0.1542	0.1610	0.1647	0.1588	0.1497	0.1363
Win/loss	N/A	15/4	18/1	17/2	17/2	16/3	16/3	15/4

4.1 Parameters

The parameter setting are showing below. The fb, fbDocs, fbTerms and fbMu is setting as the handout write. And I set mu to 1000 and lambda to 0.7 as baseline for Indri. Because large lambda means less smoothing and better for the short query and all of our queries are short, so I set lambda to 0.7 for baseline. I set mu to 1000 because it gives a normal performance for Indri model, not too high MAP&P@n and not too low MAP&P@n, it is convenient for us to compare it with results under different mu later. Also, the large mu is more important to short documents and our retrieval documents are mostly long documents, so I set it to a small number, like 1000. . And set these parameter as same as baseline is convenient to compare the result. And for fbOrigWeights, I set it from 0.0 to 1.0 as the table shows.

RankedBoolean AND	trecEvalOutputLength=100
Reference of BOW	Nothing to set

Reference System of query expansion	trecEvalOutputLength=100
	Indri:mu=1000
	Indri:lambda=0.7
	fb=true
	fbDocs=10
	fbTerms=10
	fbMu=0
	fbOrigWeight=0.0, 0.2, 0.4, 0.6, 0.8, 1.0

	fbInitialRankingFile=Indri-Bow.teIn

4.2 Discussion

The fbOrigWeight determine how important original query in the expansion query. When fbOriginWeight euqals 1.0, that means we only use original query, so we get same result as our own system.

When the fbOrigWeight is 0, that means we only use expansion query to retrieve documents. The precision is low and it is reasonable. As we can see from the expansion query, like query "705: Iraq foreign debt reduction", the expansion query contains lots of meaningless query terms like "we" and "country", which contributes nothing to the relevant results and have high weight. And "749: Puerto Rico state", whose expansion query includes "1951" and "1952" as query terms, obviously it should be treat as stop words. In this situation, the retrieval results will be misleading to un-relevant documents. But the MAP is higher than BOW of Indri. Since we use top N score terms in top M documents as expansion query terms (whose score is related to its tf and ctf), it will help to get more relevant documents. Query expansion is very useful in high recall needed situation. As for WIN/LOSS, since we have very high recall and not too bad precision for top P@N, it could be high for each query MAP and average MAP.

And as the fbOriginWeight increase from 0 to 0.8, the P@N increase then decrease a little but the MAP decrease. That's quiet make sense. When the fbOriginWeight increase, the original query contributes more and there will be less weight assigned to un-relevant query terms which produced by expansion query. So, the results will be more relevant in the P@N. The best number of original weights various by query. In this situation, the best fbOrignWeight is 0.4.

At the same time, the MAP reduce to the same as BOW of Indri as fbOriginWeight increase. This because the less weight assigned to expansion query, which lead to lower recall. As for WIN/LOSS, the expansion query's WIN decrease and LOSS increase. Since the recall decrease a lot as original weight increase, the WIN/LOSS results is expected.

Some queries are quiet unstable during the change of original weight. For example, the query "705:Iraq foreign debt reduction", query "707:Aspirin cancer prevention", query "723:Executive privilege" and "749:Puerto Rico state", their MAP changed huge. This may because the information need is ambiguous and these query terms' quality is low. So the expansion query terms is also low quality, like expansion query of "749:Puerto Rico state", high weight gives to terms like "1951", "1952", "pfc", "pvt", "action", "kill", "army" and "municipality". Half of query terms in expansion query is meaningless and can't help get relevant results. What's worse? It will help get more totally un-relevant documents. Some query are quiet stable, like query "710:Prostate cancer treatments". This may because the query terms are related closely to each other.

5 Experiment 5: Smoothing on longer queries

	Indri BOW, Reference System	Query Expansion, fbTerms = 10					
		μ					
		500	1500	2500	3500	4500	5500
P@10	0.6000	0.5579	0.6211	0.6000	0.5895	0.5842	0.5737
P@20	0.5447	0.5421	0.5447	0.5447	0.5342	0.5211	0.5132
P@30	0.5228	0.4912	0.5228	0.5298	0.5211	0.5175	0.5035
MAP	0.1483	0.1550	0.1530	0.1517	0.1484	0.1434	0.1395
Win/loss	15/4	16/3	16/3	16/3	16/3	16/3	16/3

	Indri BOW, Reference System	Query Expansion, fbTerms = 20					
		μ					
		500	1500	2500	3500	4500	5500
P@10	0.6000	0.5684	0.6105	0.6158	0.5947	0.6053	0.5737
P@20	0.5447	0.5395	0.5395	0.5421	0.5316	0.5158	0.5184
P@30	0.5228	0.4982	0.5175	0.5193	0.5211	0.5228	0.5123
MAP	0.1483	0.1565	0.1539	0.1505	0.1464	0.1421	0.1392
Win/loss	15/4	16/3	16/3	16/3	15/4	15/4	15/4

	Indri BOW, Reference System	Query Expansion, fbTerms = 30					
		μ					
		500	1500	2500	3500	4500	5500
P@10	0.6000	0.5895	0.6158	0.5947	0.6053	0.6000	0.5684
P@20	0.5447	0.5500	0.5421	0.5368	0.5237	0.5132	0.5105
P@30	0.5228	0.5035	0.5211	0.5175	0.5228	0.5211	0.5105
MAP	0.1483	0.1585	0.1528	0.1494	0.1451	0.1411	0.1375
Win/loss	15/4	16/3	16/3	16/3	16/3	15/4	15/4

5.1 Parameters

The parameter setting are showing below. The fb, fbTerms, fbOrigWeights and fbMu is setting as the handout write. And I set mu to 1000 and lambda to 0.7 as baseline for Indri. Because large lambda means less smoothing and better for the short query and all of our queries are short, so I set lambda to 0.7 for baseline. I set mu to 1000 because it gives a normal performance for Indri model, not too high MAP&P@n and not too low MAP&P@n, it is convenient for us to compare it with results under different mu later. And for fbDocs, I set it as 40 based on the experiment 2 and experiment 3. Mu in Indri is changing from 1500 to 6500.

Reference of BOW	Nothing to set
Reference System of	trecEvalOutputLength=100

query expansion	Indri:lambda=0.7
	fb=true
	fbDocs=40
	fbTerms=10
	fbMu=0
	fbOrigWeight=0.7
	fbInitialRankingFile=Indri-Bow.teIn

5.2 Discussion

In general, a large mu is more important for short document and less important for long document, so decrease mu have more influence on the short document. The short document which is relevant may rank low now. So the precision goes down for P@10, P@20 and P@30 as mu increase. Comparing to the baseline, the MAP down as the mu increase. As the mu goes up, it will help short documents to get a high score and the “small samples” and “unseen words” are all detected. More relevant documents will be retrieval and recall goes up. It is good in a range. But when goes out of this range and put too much importance on the short documents will cause the long relevant document which includes more informative terms ranking low. So MAP decrease as precision decrease.

As we can see from the results, when fbTerms is 10, as the mu increase, the precision and recall decrease very fast compare to fbTerms 20 and fbTerms 30 and reaches highest when fbTerms is 1500. Since the mu is related to the document length, large mu is important to short documents, that means the initial documents mostly are long documents. The same as fbTerms 20 and fbTerms 30. And for the fbTerms 20, we get best top N precision when mu is 500 and not bad MAP overall. When fbTerms is 30, we have highest MAP when mu is 500. It seems like we can use the same small mu for each fbTerms situation and can get good result. It is reasonable, since mu is corresponded to the document length, and we have a lot of long documents, we can use small mu to get good results.

Comparing MAP and precision between different fbTerms under same mu, we can find that MAP under fbTerms 10 has highest precision when mu larger than 2500. This may because long expansion query do not need much smoothing like short expansion queries. For fbTerms=10 and fbTerms=20, we find that P@N changes a lot when mu larger than 500 and less than 5500, thought the MAP reduce huge. For fbTerms=30, we find that P@N is quiet stable when mu larger than 500 and less than 5500, thought the MAP reduce huge. That means large smoothing number is not that important to the long expansion query but may important to short expansion queries.

Comparing MAP and precision between baseline and values under different fbTerms, we can see that when fbTerms=10, mu less than 3500 can have a better performance than baseline, and this value is 2500 for fbTerms=20 and fbTerms=30. And for fbTerms=30, theperformance is quiet near to baseline when mu equals 2500. This phenomenon indicates that we could use a smaller μ to do the smoothing for longer expanded queries. The longer expanded queries require less smoothing than the shorter unexpanded queries.