Your Name: Jiangtian Qian

Your Andrew ID: jiangtiq

Homework 2

Collaboration and Originality

Your report must include answers to the following questions:

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: Jiangtian Qian

Your Andrew ID: jiangtiq

Homework 2

Instructions

1 Experiment 1: Baselines

	Ranked Boolean	**BM25 BOW**	0	**Indri BOW**
P@10	0.0900	0.5100	0.6300	0.6600
P@20	0.1300	0.5000	0.5800	0.5700
P@30	0.1267	0.4500	0.5467	0.5333
MAP	0.0104	0.1303	0.1539	0.1526

2 Experiment 2: BM25 Parameter Adjustment

2.1 k_1

	k_1							
	0	0.2	0.5	1.2	1.8	2	3	4
P@10	0.1800	0.5100	0.5300	0.5100	0.5100	0.5100	0.4600	0.4600
P@20	0.1800	0.4800	0.4750	0.5000	0.5150	0.4900	0.4450	0.4300
P@30	0.2000	0.4200	0.4300	0.4500	0.4600	0.4533	0.4500	0.4300
MAP	0.0404	0.1247	0.1289	0.1303	0.1282	0.1230	0.1132	0.1074

.

2.2 b

	b							
	0.75	0.85	1.00	0.65	0.55	0.45	0.35	0
P@10	0.5100	0.5100	0.3500	0.5400	0.6000	0.6400	0.6300	0.3600
P@20	0.5000	0.5150	0.3550	0.5300	0.5650	0.6100	0.6000	0.3700
P@30	0.4500	0.4600	0.3367	0.4733	0.5033	0.5300	0.5500	0.3567
MAP	0.1303	0.1295	0.0996	0.1447	0.1522	0.1611	0.1640	0.0908

2.3 Parameters

The weight for BM25 model has the function as:

$\sum (log \frac{N - df_t + 0.5}{df_t + 0.5}) \frac{tf_{t,d}}{tf_{t,d} + k_1((1-b) + b \frac{doclen_d}{avg_d oclen})} \frac{(k_3 + 1) qtf_t}{k_3 + qtf_t}$, k3 is always 0. We use $log \frac{N - df_t + 0.5}{df_t + 0.5}$ to tell whether this document includes lots of information or a common one, we call it IDF. The rarer the more important and will have high weight as IDF. But it will be effect if the document is too long, so we use parameter k1 and b here. First, we set b as 0.75 and change k1's value as shown above. k1 is responsible for calibrates the documents frequency scaling. Large k1 means using raw frequency of the term and k1 equals 0 means use no frequency, which is a binary model. That means, as k1 increase, the weight will decrease for certain term. At first, if this term is rare in the term collection, it will have a large weight. As k1 increase, the weight will be decrease more for rare term than the high tf term. So, it will decrease the difference between rare (informative) term and normal (less informative) term. The relevance between query and documents may be effect more by the less informative term. So, the top N result may have more documents not that relevant than before. The MAP may decrease and the precision may decrease a little. So, I choose k1 start from the lowest 1.2 and increase it to see if there any trends to indicate what I think.

Parameter b should be larger than 0 and less than 1 for the scaling goal. Parameter b is determining the relevance by document length. The very large b, like b = 1 corresponds to scaling the term weight by the document length, which means long documents has more effect on relevance between query. The very small b, like b = 0, corresponds to no normalization, which means long document has less effect on relevance between query. So, I choose some larger b like 0.85. Since the largest b is 1, so I choose smaller b which less than 0.75 to see the trends of precision and MAP.

2.4 Discussion

k3 = 0 means query term frequency has no effects.

K1:

As we can see above, the precision for P@10 is stable when k1 increase because the top 10 result has a large weight and informative terms' effect is not that huge compare to the weight. The top 20 and top 30 precision is unstable increase and decrease, but the main trend is increase as the k1 increase. This because as k1 increase, the weight of rare term gains more importance. If the rare query terms truly have important information, the relevant document will rank high. As the precision change, that means rare terms have effect on the result return. And when we set k1 to 0, the precision and MAP is extremely low. That because we use the raw frequency for the weight computation. When we use the raw frequency, the weight is determined by the document frequency, which means terms like stop words have more effect informative terms. So the non-relevant documents will rank high. And the MAP is decrease as k1 decrease. This because recall is decrease for we get less documents.

b:

As b decrease, precision of P@10, P@20 and P@30 all increase. B decrease means putting less document length influence on the scoring. In this case, the long documents, which has lots of meaning less words, will have less influence in to determine the rank of itself. For the same tf, when document is longer, the more possibility to have query term. When b equals 0 or b equals 1, the precision is extremely low compare to others. When b = 1, that means document length has more impact on the scoring. The longer documents may contain more query terms and has more effect on the query. But long documents may contain the meaningless query term and get high score, which will reduce the relevant documents in the top N. When b = 0, document length has no effect on the ranking, the long document has more possibility to rank high, which may not that relevant. It is better for the long document. The MAP increase as b decrease, because when b decrease, the precision increase as we analyzed above.

3 Experiment 3: Indri Parameter Adjustment

3.1 μ

	μ							
	0	1000	2400	2500	3000	4000	5000	6000
P@10	0.4100	0.6100	0.6500	0.6600	0.6500	0.6500	0.6200	0.6200
P@20	0.3900	0.5850	0.5700	0.5700	0.5700	0.5550	0.5450	0.5350
P@30	0.3500	0.5233	0.6333	0.5333	0.5333	0.5300	0.5033	0.4867
MAP	0.1063	0.1474	0.1543	0.1526	0.1519	0.1482	0.1457	0.1423

3.2 λ

	λ							
	0	0.2	0.3	0.4	0.5	0.6	0.7	1
P@10	0.6300	0.6500	0.6500	0.6600	0.6400	0.6300	0.6000	0.6500
P@20	0.5800	0.5850	0.5750	0.5700	0.5600	0.5650	0.5500	0.5750
P@30	0.5467	0.5433	0.5333	0.5333	0.5267	0.5033	0.5000	0.5333
MAP	0.1539	0.1515	0.1525	0.1526	0.1533	0.1509	0.1482	0.1528

3.3 Parameters

The query score for each qi is $p(q_i|d) = (1-\lambda)\frac{tf_{q_i,d} + \mu p_{MLE}(q_i|C)}{length(d)+\mu} + \lambda p_{MLE}(q_i|C)$.

For mu, the very small mu, like 0 means there is no smoothing, we only use maximum likelihood to get the score for query q. So, there is less smoothing for small mu. For a large mu, it means more smoothing for query term frequency in the collection add more influence in the scoring. So, we set mu to 0 at first to see if the precision and MAP is the worst. And then I increase mu from a small number 2300 to a large number to observe the change.

For lambda, the small lambda means more smoothing for it compensate for differences in the word model. Larger lambda puts less importance on the idf weighting, it is better for the short query. And large

lambda means less smoothing. So I set mu to 2500 and lambda to 0 and 1 to see the performance in the extreme situation. And then I increase lambda from a very small value 0.3 to a larger value to see the change of the precision and MAP.

3.4 Discussion

mu:

When mu is very small, we can see both precision and MAP's value is very small. As we can see from this equation $p(q_i|d) = (1-\lambda)\frac{tf_{q_i,d} + \mu p_{MLE}(q_i|C)}{length(d)+\mu} + \lambda p_{MLE}(q_i|C)$. When mu is very small, we are more close to only use maximum likelihood to get the score for query q. And 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. Comparing to the baseline, the MAP goes up and than 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. And Indri is stable when mu changes a lot.

lambda:

$$p(q_i|d) = (1-\lambda)\frac{tf_{q_i,d} + \mu p_{MLE}(q_i|C)}{length(d)+\mu} + \lambda p_{MLE}(q_i|C)$$

As we can see from the table above, when lambda = 0, the P@10 is low but P@20 and P@30 is high. Small lambda is good for short query, since maximum likelihood of the query in the collection will put less influence on the scoring, "idf" weighting is less important. And our queries here are all short, like 3 to 4 words, so every query terms must match. So lambda = 0 will give us the best precision and MAP. As the lambda increase, it is good for long query and bad for short query. Because long query has many query terms so most query terms much is ok, "idf" weight is more important. So, the precision and MAP decrease as lambda increase. And when lambda = 1, we get a very good precision and MAP. Because at this time, query score is depend on the P(qi|C) and our query here are all short query and every words is meaningful, so we will get good precision and MAP only use P(qi|C).

4 Experiment 4: Different representations

4.1 Example Query

#AND (indiana.body child.body support.body)
#AND (#WSUM(0.2 indiana.url 0.3 indiana.title 0.5 indiana.body) #WSUM(0.2 child.url 0.3 child.title 0.5 child.body) #WSUM(0.2 support.url 0.3 support.title 0.5 support.body))
#AND (#WSUM(0.2 indiana.keywords 0.3 indiana.keywords 0.5 indiana.body) #WSUM(0.2 child.keywords 0.3 child. keywords 0.5 child.body) #WSUM(0.2 support.keywords 0.3 support. keywords 0.5 support.body))
#AND (#WSUM(0.2 indiana.url 0.3 indiana.keywords 0.5 indiana.body) #WSUM(0.2 child.url 0.3 child. keywords 0.5 child.body) #WSUM(0.2 support.url 0.3 support. keywords 0.5 support.body))
#AND (#WSUM(0.1 indiana.url 0.2 indiana.keywords 0.3 indiana.title 0.4 indiana.body) #WSUM(0.1 child.url 0.2 child. keywords 0.3 child.title 0.4 child.body) #WSUM(0.1 support.url 0.2 support. keywords 0.3 support.title 0.4 support.body))
#AND (#WSUM(0.1 indiana.url 0.1 indiana.keywords 0.1 indiana.title 0.7 indiana.body) #WSUM(0.1 child.url 0.1 child. keywords 0.1 child.title 0.7 child.body) #WSUM(0.1 support.url 0.1 support. keywords 0.1 support.title 0.7 support.body))

4.2 Results

	Indri BOW (body)	**0.20 url 0.00 keywords 0.30 title 0.50 body**	**0.00 url 0.20 keywords 0.30 title 0.50 body**	**0.20 url 0.30 keywords 0.00 title 0.50 body**	**0.10 url 0.20 keywords 0.30 title 0.40 body**	**0.10 url 0.10 keywords 0.10 title 0.70 body**
P@10	0.6600	0.6000	0.6100	0.6300	0.5900	0.6400
P@20	0.5700	0.5400	0.5250	0.5400	0.5150	0.5550
P@30	0.5333	0.4867	0.4867	0.5067	0.4767	0.5100
MAP	0.1526	0.1469	0.1466	0.1486	0.1443	0.1511

4.3 Weights

Query terms can be anywhere, may in url, keywords, title and body only or in this place at the same time. First, I test the precision and MAP for the default Indri BOW as baseline. I set large value for body for several reasons. First, the body part usually contains the most words and longer than our field, the query terms has the highest possibility to appear in body. Second, our queries here are more likely to appear in body than other field, for they are almost phrase and descriptions. Third, the body is most effective to describe user's information need. Then I keep body weight no change and iterate setting the url, keywords and title to be 0. The keywords and title may have more chance to contain query terms than url, for url is short and sometimes contains meaningless character, like ?+ and so on. So, I set the minimum weight for url. At last, I set all the url, keywords and title to 0.1 and body to 0.7 to see the body's influence on the

final score. And to see if we cover all the field, can we increase the precision and MAP. If the score is high, my hypothesis that body is the most important part and should be set highest weight is true.

4.4 Discussion

As we can see from the table above, the default query has the highest P@10, P@20, P@30 and MAP. This because body part usually contains the lots of words and longer than other fields. So the query terms has the highest possibility to appear in body. The "Decorative slate sources Prostate cancer treatments Census data applications Hubble telescope repairs whales save endangered Gastric bypass complications Airline overbooking hedge funds fraud protection Puerto Rico state Scrabble Players" may appear in title, url and keywords, but body part has higher possibility to contains them. So, it will ensure us to find them instead of set high weight in other fields.

Compare the second, third and fourth data in the table, the structured query with higher keywords weight has highest precision and MAP. Because most of these query terms are nouns and very possible to be keywords for documents. Keywords always present the main idea of the documents, so if query terms included in keywords, the document is possibly talking about this query term, which means the document is relevant. So the precision is higher than include other field. And the last column of data is much higher than than previous columns, because we add body field. Talking about url, it may include query terms but talking about other things, which means the document is non-relevant. The last column has much better performance than the previous columns, because we assign more weight to body. And we can conclude from this result that body should be assign more weight because the information need type is different and information need are ambiguous sometimes.

5 Experiment 5: Sequential dependency models

#wand(0.5 #and(indiana child support) 0.25 #and(#near/1(indiana child) #near/1(child support)) 0.25 #and(#window/8(child support) #window/8 (indiana child)))
#wand(0.5 #and(indiana child support) 0.25 #and(#near/1(indiana child) #near/1(child support)) 0.25 #and(#window/8(child support) #window/8 (indiana child)))
#wand(0.6 #and(indiana child support) 0.1 #and(#near/1(indiana child) #near/1(child support)) 0.3 #and(#window/8(child support) #window/8 (indiana child)))
#wand(0.7 #and(indiana child support) 0.1 #and(#near/1(indiana child) #near/1(child support)) 0.2 #and(#window/8(child support) #window/8 (indiana child)))
#wand(0.4 #and(indiana child support) 0.3 #and(#near/1(indiana child) #near/1(child support)) 0.3 #and(#window/8(child support) #window/8 (indiana child)))
#wand(0.6 #and(indiana child support) 0.3 #and(#near/1(indiana child) #near/1(child support)) 0.1 #and(#window/8(child support) #window/8 (indiana child)))

5.1 Example Query

5.2 Results

	Indri BOW (body)	0.50 AND 0.25 NEAR 0.25 WINDOW	0.60 AND 0.10 NEAR 0.30 WINDOW	0.70 AND 0.10 NEAR 0.20 WINDOW	0.40 AND 0.30 NEAR 0.30 WINDOW	0.60 AND 0.30 NEAR 0.10 WINDOW
P@10	0.6600	0.4300	0.4800	0.4533	0.4400	0.4200
P@20	0.5700	0.4150	0.4850	0.5000	0.3950	0.3950
P@30	0.5333	0.3733	0.4433	0.4700	0.3433	0.3533
MAP	0.1526	0.1170	0.1308	0.1362	0.1104	0.1150

5.3 Weights

User's information need always ambiguous and has lots of different types. For the queries we have, we can figure out that "Puerto Rico", "Prostate cancer" and "Airline overbooking" are phrase like but they are small part of the whole queries. So I set low value to the Near weight. And we can see that terms in "Hubble telescope", "Puerto Rico", "Airline overbooking", "Decorative slate" and "Census data" can be close to each other because only they combined together, they can express a certain meaning and sentence like.

So, I set a not too high but not too low weight to Window. And I set around 0.5 value to AND and because I want to make sure to find things I want and not that much of queries are sentence like or phrase like. At last, I want to compare how near and window influence the whole precision and MAP, so I set the same AND and 0.1 and 0.3 for Window and Near respectively / and 0.3 and 0.1 for Window and Near respectively.

5.4 Discussion

As we can see, the baseline has the best performance in P@10, P@20, P@30 and MAP.

Although we have some terms can be phrase, but in the sequential dependency model, it set all the near query terms as phrase, which not only increase the running time, but also reduce precision as we may retrieval documents contain meaningless query terms.

For example, the near operator for "Census data applications" is Near/1 (Census data) Near/1(data applications). The user want the documents contain "Census" data applications, the application and data is less important than "Census". But for the Near/1 operator, the documents returned may only contain "data application", which has no relationship with "Census". This results will drive non-relevant documents to rank high.

And we can see clearly with comparison between the third column and the last column. When we set Window low and Near high, the precision and MAP decrease, especially the precision. That means we have more sentence like query and less phrase query here. Because the informational information needs are rare, information needs are often navigational and transactional. And when we set AND to 0.4, Window to 0.3 and Near to 0.3, we have extremely low precision and MAP here. That proves the Near and Near problem I mentioned above.

But, when I set AND to 0.7, Near to 0.1 and Window to 0.2, we have worse precision compare to AND to 0.6, Near to 0.1 and Window to 0.3. That because we give more weight to sentence like query and we truly have a lot of sentence like query here, like “Hubble telescope”, “Puerto Rico”, “Airline overbooking”, “Decorative slate” and “Census data”.