Your Name: Sz-Rung Shiang

Your Andrew ID: sshiang

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

2. Did you give help of any kind to anyone in developing their software for this assignment (Yes or No)?

   No

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

4. Are you the author of every word of your report (Yes or No)?

   Yes

Your Name: Sz-Rung Shiang

Your Andrew ID: sshiang

Homework 2

Instructions

1 Experiment 1: Baselines

	Ranked Boolean	BM25 BOW	Indri BOW
P@10	0.2000	0.2600	0.2400
P@20	0.2650	0.3300	0.4100
P@30	0.2833	0.3633	0.4233
MAP	0.1015	0.1881	0.2084

2 Experiment 2: BM25 Parameter Adjustment

2.1 k_1

	k_1							
	0.1	0.9	1.0	1.1	1.2	1.3	1.4	2.0
P@10	0.2600	0.2600	0.2600	0.2600	0.2600	0.2600	0.2600	0.2700
P@20	0.3350	0.3300	0.3300	0.3300	0.3300	0.3300	0.3300	0.3250
P@30	0.3700	0.3700	0.3700	0.3633	0.3633	0.3633	0.3633	0.3600
MAP	0.1885	0.1885	0.1884	0.1883	0.1881	0.1881	0.1881	0.1879

.

2.2 b

	b							
	0.125	0.25	0.375	0.5	0.75	1.0	1.25	1.5
P@10	0.4200	0.3200	0.3100	0.2700	0.2600	0.2100	0.0090	0.0080
P@20	0.4650	0.4350	0.3700	0.3650	0.3300	0.2800	0.1200	0.0085
P@30	0.4800	0.4567	0.4400	0.4033	0.3633	0.3167	0.1600	0.0080
MAP	0.2394	0.2240	0.2088	0.1994	0.1881	0.1578	0.0904	0.0569

2.3 Parameters

According to Wikipedia page, most of the papers choose k_1 from 1.2 to 2.0; as a result, I explored the parameters within this range. After finding out there is not much difference, I tried to use the value farer away (0.1 in the table and 10 in the following discussion part) from this range to see the tendency of the performance. For parameter *b*, because the range should be [0,1] to prevent any negative weighting for term frequency, I tried the parameter within this range. In addition, the smaller value has better performance, so I tried smaller spacing for smaller *b*. The parameter b larger than 1.0 should be not effective and harmful, to verify this, I tried to use these values.

2.4 Discussion

k_1 is the shrinkage parameter for the term frequency. For example, document with term frequency as 10 doesn't mean that it's important 10 times more than the document with term frequency as 1. As *k1* becomes smaller, the shrinkage effect becomes more obvious. In the experiment part, the difference of performance under different k_1 is not significantly different. The reason might be that most of the term frequency is 1 or 2, and the variance of term frequency is not large; therefore, the factor k_1 doesn't affect the performance. The additional experiment is for k_1 as 10, the performance drops significantly from MAP 0.1881 to 1590, which means that too much term frequency shrinkage is harmful.

b is the parameters for normalizing document length. If *b* is fixed, when the document is longer, *b**|D|/avgLen is larger, thus the weighted term frequency is smaller. As *b* becomes larger, there is more effect on penalizing longer documents, while as *b* is close to 0, the weighted term frequency is close to term frequency. In the experiments part, however, smaller *b* has better performance. *b*=0.125 has the best performance and when *b* gets larger, both precision and MAP deteriorates. It may indicate that the variance of the length of document in the documents set is not so significant.

Another part is that b shouldn't be more than 1.0; otherwise (1-b+b*|D|/avgLen) will be negative for longer documents and the weighted term frequency will be negative, which doesn't make sense. As we can see from the experiments that when b is 1.25 and 1.5, the precision and MAP becomes very low due to the negative weighted term frequency.

3 Experiment 3: Indri Parameter Adjustment

3.1 µ

	µ							
	2000	2500	3000	4000	5000	6000	7000	8000
P@10	0.2300	0.2400	0.2900	0.3000	0.3200	0.3200	0.3500	0.3500
P@20	0.3850	0.4100	0.4200	0.4150	0.3950	0.4150	0.4400	0.4400
P@30	0.4133	0.4233	0.4233	0.4300	0.4567	0.4600	0.4567	0.4567
MAP	0.2041	0.2084	0.2137	0.2165	0.2182	0.2201	0.2233	0.2236

3.2 λ

	λ							
	0.05	0.1	0.2	0.3	0.35	0.4	0.45	0.5
P@10	0.2500	0.2500	0.2400	0.2400	0.2400	0.2400	0.2400	0.2100
P@20	0.4100	0.4100	0.4100	0.4100	0.4100	0.4100	0.4100	0.4100
P@30	0.4267	0.4267	0.4233	0.4233	0.4233	0.4233	0.4200	0.4200
MAP	0.2091	0.2088	0.2089	0.2085	0.2086	0.2084	0.2083	0.2078

3.3 Parameters

The parameter *mu* a smoothing term for document length and document prior. If the number is too small then the value will be close to using only term frequency. On the other hand, if the number is too large, then mu*P_{MLE} term might be larger than term frequency and mu will be dominant for the term of document length; as a result, the value for this term will not vary for any term frequency or document length. I tried the value 2000 and 3000, which are near to 2500, and then I choose the tendency that can get higher performance.

The parameter *lambda* is the balancing term, thus it should be between 0 and 1; otherwise the term frequency or the smoothing term from maximum likelihood estimator might be negative. I start exploring the parameter from the neighboring of 0.4, which is 0.5 and 0.3. As we can see that the performance for *lambda* as 0.5 is worse; therefore I didn't exploring any value larger than it.

3.4 Discussion

The parameter *mu* a smoothing factor especially for document length. As the value gets higher, the longer documents can get more benefits to compete with shorter documents. In our experiment part, when *mu* is 8000 the performance is the best, while when *mu* is 2000 the performance is the worst. The experimental results accord with the previous result in section 2.2. The observation is that in our documents set, we don't like to penalize for variance of the document length (especial for longer documents).

The parameter *lambda* is for the weighting between term frequency and the smoothing factor using maximum likelihood estimator. As *lambda* gets higher, the smoothing effect becomes more significant. In the experiment part, the smaller lambda achieves the best performance. There are might be two reasons for that. First, the number of relevant documents with all the query terms in the documents is high enough; therefore, doing retrieval without smoothing might be fine. Second, the length of query is short in our case (1-3 terms per query), which means missing one of any terms might be crucial. As a result, using smoothing might not be important in our case.

4 Experiment 4: Different representations

4.1 Example Query

The following is the example query for "indiana child support" with field weights equally (0.20, 0.2, 0.2, 0.2, 0.2) for each field (url, keywords, title, body, inlink).

104: #AND (#WSUM(0.2 indiana.url 0.2 indiana.title 0.2 indiana.inlink 0.2 indiana.body 0.2 indiana.keywords) #WSUM(0.2 child.url 0.2 child.title 0.2 child.inlink 0.2 child.body 0.2 child.keywords) #WSUM(0.2 support.url 0.2 support.title 0.2 support.inlink 0.2 support.body 0.2 support.keywords))

104: #AND(#WSUM(0.1 indiana.url 0.1 indiana.keywords 0.1 indiana.title 0.6 indiana.body 0.1 indiana.inlink) #WSUM(0.1 child.url 0.1 child.keywords 0.1 child.title 0.6 child.body 0.1 child.inlink) #WSUM(0.1 support.url 0.1 support.keywords 0.1 support.title 0.6 support.body 0.1 support.inlink))

4.2 Results

	Indri BOW (body)	**0.20 url 0.20 keywords 0.20 title 0.20 body 0.20 inlink**	**0.10 url 0.10 keywords 0.10 title 0.60 body 0.10 inlink**	**0.10 url 0.10 keywords 0.35 title 0.35 body 0.10 inlink**	**0.05 url 0.10 keywords 0.10 title 0.70 body 0.05 inlink**	**0.10 url 0.05 keywords 0.05 title 0.70 body 0.10 inlink**
P@10	0.2400	0.2300	0.2200	0.1800	0.2200	0.2300
P@20	0.4100	0.2500	0.3500	0.2500	0.3700	0.3650
P@30	0.4233	0.2600	0.3967	0.3300	0.4000	0.4033
MAP	0.2084	0.1201	0.1866	0.1466	0.1899	0.1923

4.3 Weights

I set the sum of the weights as 1 in order to see the contribution of each filed. In the previous experiments in homework 1, using filed *body* is the most effect way in our retrieval system. I tried to use equal weights for each fields to justify it, and I got a worse result as expected. Then I tried to use different weights for other fields but keeping the weight for body highest.

4.4 Discussion

In the experiments, the field *body* outperforms any other fields in the retrieval system. The reason might be that if there is term that is not in the documents, but Indri operator still provides the default value for those terms. When the default value is large enough, even the documents with missing query terms matching might have higher score than other documents with every query terms matching (but may have lower term frequency). Recall in homework 1 *AND* operator also performs better than *OR* operator does. It's to say that the number of relevant documents is enough, and what we have to do is to distinguish the documents with all terms matched. Other fields like *url*, *title*, *inlink* and *keywords* are vulnerable to the missing matching terms.

5 Experiment 5: Sequential dependency models

5.1 Example Query

The following is the example query for “Indiana child support” with weights for AND, NEAR and Window as 0.5, 0.25 and 0.25 respectively.

104:#wand(0.5 #and(indiana child support) 0.25 #and(#near/1(child support) #near/1(indiana child)) 0.25 #and(#window/8(child support) #window/8(indiana child)))

104:#wand(0.9 #and(indiana child support) 0.05 #and(#near/1(child support) #near/1(indiana child)) 0.05 #and(#window/8(child support) #window/8(indiana child)))

104:#wand(0.7 #and(indiana child support) 0.2 #and(#near/1(child support) #near/1(indiana child)) 0.1 #and(#window/8(child support) #window/8(indiana child)))

5.2 Results

	Indri BOW (body)	**0.50 AND 0.25 NEAR 0.25 WINDOW**	**0.90 AND 0.05 NEAR 0.05 WINDOW**	**0.80 AND 0.10 NEAR 0.10 WINDOW**	**0.70 AND 0.20 NEAR 0.10 WINDOW**	**0.70 AND 0.10 NEAR 0.20 WINDOW**
P@10	0.2400	0.2300	0.2500	0.2500	0.2500	0.2500
P@20	0.4100	0.3850	0.4000	0.3900	0.3900	0.3900
P@30	0.4233	0.4067	0.4167	0.4033	0.4133	0.4167
MAP	0.2084	0.1889	0.2077	0.1970	0.1931	0.1971

5.3 Weights

The sum of weights is set to be 1. For baseline, the weight is (1.0, 0.0, 0.0) for only using AND operator. To see whether it’s good to have other operators, I choose (0.5, 0.25, 0.25) but finding out the performance is worse. As a result, the strategy for choosing parameters is to make weight of AND higher and to distinguish NEAR and WINDOW which is better.

5.4 Discussion

In table 5.2, there are two observations for this experiment. First, AND operator is the most effective way for our retrieval system. In addition, it’s the most efficient way that AND operator doesn’t have to consider the position of terms in the documents. Second, WINDOW/8 performs better than NEAR/1. Near/1 is a very strict operator that makes the constraints the terms are adjacent; however, some queries might not be phrases or phrases that allow other terms in it. The experimental results for using only NEAR/1 and WINDOW/8 are 0.1600 and 0.1890 in MAP, that is to say, the goodness of each operator is AND > WINDOW/8 > NEAR/1. In conclusion, the stricter the operator is, the worse the performance.

For computation efficiency, AND > NEAR/1 > WINDOW/8. In my implementation, Window doesn’t care the order of terms and therefore I made the *for* loop check every pair of terms, thus it takes more time than NEAR/1 operator.