Your Name: Han Zhang

Your Andrew ID: hanzhan2

Homework 2

Collaboration and Originality

Your report must include answers to the following questions:

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: Han Zhang

Your Andrew ID: hanzhan2

Homework 2

1 Experiment 1: Baselines

	Ranked Boolean	BM25 BOW	Indri BOW
P@10	0.04	0.37	0.49
P@20	0.08	0.355	0.425
P@30	0.0867	0.34	0.3833
MAP	0.0079	0.0614	0.0973

2 Experiment 2: BM25 Parameter Adjustment

2.1 k_1

	k_1							
	0.001	0.01	0.025	0.05	0.1	1.2	5	100
P@10	0.44	0.44	0.44	0.44	0.45	0.37	0.31	0.07
P@20	0.43	0.435	0.435	0.43	0.435	0.355	0.285	0.07
P@30	0.3967	0.3967	0.3967	0.3967	0.41	0.34	0.2733	0.1067
MAP	0.0875	0.0876	0.0868	0.0875	0.07868	0.0614	0.0437	0.009

.

2.2 b

	b							
	0	0.1	0.13	0.15	0.2	0.5	0.75	1
P@10	0.35	0.48	0.48	0.48	0.49	0.48	0.37	0.24
P@20	0.38	0.44	0.445	0.45	0.46	0.415	0.355	0.25
P@30	0.3567	0.4167	0.4233	0.4333	0.433	0.38	0.34	0.27
MAP	0.0865	0.1063	0.1082	0.1074	0.1056	0.08	0.0614	0.0453

2.3 Parameters

K1 is the term weighting function. It is monotonic. The bigger k1 is, the smaller marginal effect for high term frequency. I would like to see the performances of a very small k1(0.001) and a very big k1(100) first. Then I generated a hypothesis that a big k1 performs badly, and from the nearly 0 to a small k1, there will be a peak MAP. Thus, I picked some k1s from 0.001 to 0.1 to see the trend.

According to the role of b, at first, I thought there will be a good performance in the middle values, which means the average normalization performs good. So I tried 0.5 and it performed much better than default.

Then I tried extreme values 0 and 1. The result showed that 0 performed even better than 0.5. Therefore, I thought there will be a decreasing trend from 0 to middle and I many values from 0 to 0.5.

2.4 Discussion

For k1, I thought there will be a peak between 0.001 and 0.1, but the result showed that there is a small bottom at 0.025 and the performances of all selected values didn't have too much difference. In another word, these small k1 performs similarly well. Generally, there is a decreasing trend from the nearly 0 to larger value. And values from 0.001 to 0.05 perform similarly well. K1 is used to reduce the term frequency's positive effect on score when term frequency becomes large. The reason why a low value k1 performs better is that based on the Zipf's law, the majority words in a collection has a low frequency lower than 2 and half of the vocabulary only show once. The lower k1 is, the faster the tf/(tf+k1*constant) grows when tf increases from 0 to 1 to 2. Consequentially, this difference give credit to a low effective k1.

B is used to normalize the document length and reduce the possible benefit of long passage. The bigger b is, the more influence of the document length has. The experiment result shows that from 0 to 1, the MAP increased a little at the small values and decreased after that, which matched my thought. In my opinion, this is because that the length of document should be taken care, but it is sensitive and should not be focused.

3 Experiment 3: Indri Parameter Adjustment

3.1 μ

	μ							
	1000	2000	2500	3500	5000	6500	8500	10000
P@10	0.44	0.46	0.49	0.47	0.46	0.46	0.41	0.39
P@20	0.41	0.43	0.425	0.425	0.43	0.41	0.41	0.405
P@30	0.3767	0.3933	0.3833	0.3967	0.4	0.4	0.3833	0.38
MAP	0.0796	0.936	0.0973	0.0965	0.0948	0.0935	0.0887	0.0871

3.2 λ

	λ							
	0.01	0.1	0.2	0.3	0.4	0.6	0.8	0.9
P@10	0.48	0.48	0.48	0.48	0.49	0.47	0.37	0.31
P@20	0.42	0.41	0.425	0.425	0.425	0.395	0.335	0.29
P@30	0.41	0.4067	0.3967	0.3967	0.3833	0.3733	0.3333	0.29
MAP	0.0993	0.0997	0.099	0.0982	0.0973	0.0905	0.0765	0.0656

3.3 Parameters

For μ, I thought both of extreme small and extreme large value are not good, but the middle value will perform better. To test my hypothesis, I selected the value distributed uniformly to see their performance.

Based on the theories from class, the majority of query set we used is short queries so that small λ may performance better. To test that I chose the extreme values and middle value first. Since 0.6 had much better performance than 0.9, I would like to go deep in to the small values and try to find the different performances of the small λ. As a result, I selected more small λs here.

3.4 Discussion

The experiment result shows that when μ is equals to 2500, it performs best in my test. μ will help improving the estimate of the document model. It has two roles: providing the unseen words a score instead of 0 and smoothing the effect of document length. Without μ, documents that do not contain all the query terms will be scored 0, which may miss some relevant documents and reduce the precision and recall. μ helps to solve the problem. Since in the formula, μ * Pmle(qi | C) is added to term frequency, the probability is pretty small for the same reason of Zipf's Law. In order to make this part works and have some impact on term frequency, a μ that has the same magnitude with document length. 2500 may be similar to the document length in this collection.

The result shows that the small λs actually perform similarly well. λ plays the role of idf effect here. λ is used in mixture model to penalize the common words and compensate for differences in word importance. Smoothing effect decreases as λ getting close to 0. Usually, small λ with little smoothing is better for short queries and big λ with more smoothing is better for long queries. The reason that idf doesn't play an important role here may be that short queries usually have fewer common words and there is no need to penalize the words based on their frequencies since every query term matter.

4 Experiment 4: Different representations

4.1 Example Query

```
#AND(
 #WSUM(0.10 indiana.url 0.10 indiana.keywords 0.30 indiana.title 0.50 indiana.body)
 #WSUM(0.10 child.url   0.10 child.keywords   0.30 child.title   0.50 child.body)
 #WSUM(0.10 support.url 0.10 support.keywords 0.30 support.title 0.50 support.body)
)
```

Note: It is adjusted for readability. When running the program, this is a one-line query.

4.2 Results for the Query Set

	Indri BOW (body)	0.1 url 0.1 keywords 0.2 title 0.6 body	0.00 url 0.00 keywords 0.1 title 0.9 body	0.00 url 0.1 keywords 0.00 title 0.9 body	0.1 url 0.00 keywords 0.00 title 0.9 body	0.00 url 0.2 keywords 0.00 title 0.9 body
P@10	0.49	0.37	0.47	0.47	0.4	0.46
P@20	0.425	0.37	0.42	0.425	0.375	0.425
P@30	0.3833	0.3467	0.3833	0.38	0.3467	0.38
MAP	0.0973	0.0724	0.0967	0.0973	0.0763	0.0963

4.3 Weights

At first, I valued the body part the most, then title, url and keywords the least. I want to compare the general combination of them and focus on some specific fields. Thus I started with (0.1, 0.1, 0.2, 0.6) and the extreme combination of body and other fields. Then based on these results, I generated another one which weighted keywords higher to explore the differences.

4.4 Discussion

As the experiment result shows, I didn't get any weights combination that is better than body only. First the general combination (0.1, 0.1, 0.2, 0.6) got a much worse score and other combinations were better than the general combination but no better than the body only. The best one in my experiment is (0.1 keywords, 0.9 body). But when I tried to weighted keywords more, the result didn't change a lot. The result shows me that body plays the most important role and should be weighted much more than others.

The reason for this result, in my opinion is that the words used in query are some normal words and most of them will not be used in title. It is high possible that these terms in body will get a higher score than default score but these term in other field will get a default score. What is more, since these worlds will have few occurrences in other field, their field default scores will also be lower. As a result, the more weight we put on other fields, the lower score they will get. The wrong scores will lead to wrong ranking result and low precision and recall.

5 Experiment 5: Sequential dependency models

5.1 Example Query

```
#wand(

        0.3 #and( indiana child support )

        0.3 #and( #near/1( child support )  #near/1( indiana child ) )

        0.4 #and( #window/8( child support )  #window/8( indiana child ) ) )
```

Note: It is adjusted for readability. When running the program, this is a one-line query.

5.2 Results for the Query Set

	Indri BOW (body)	0.6 AND 0.2 NEAR 0.2 WINDOW	0.4 AND 0.3 NEAR 0.3 WINDOW	0.8 AND 0.2 NEAR 0.00 WINDOW	0.8 AND 0.00 NEAR 0.2 WINDOW	0.75 AND 0.00 NEAR 0.25 WINDOW
P@10	0.49	0.48	0.46	0.5	0.52	0.52
P@20	0.425	0.425	0.42	0.45	0.41	0.42
P@30	0.3833	0.4167	0.4067	0.4133	0.3967	0.41
MAP	0.0937	0.0931	0.0918	0.0891	0.0989	0.1002

5.3 Weights

Based on the results and strategies of HW1, the AND will be a good choice for general queries. So I still value AND high. In HW1, NEAR only worked for some specific combination of part of query, so I value it low. WINDOW may be a good choice since it does not care about order. I started with two general combination tests and then tested the combination of AND and NEAR or WINDOW separately. And based on the result I tested more one on AND and WINDOW.

5.4 Discussion

The experiment result aligns with my thought. The first two columns show that more weighted AND will have better precision and performance, but not that much better. (0.8 0.2 0) and (0.8 0 0.2) comparison shows that NEAR will increase the precisions with the sacrifice of recall. But WINDOW does provide a better trader off between precision and recall.

I think the reason for that is that NEAR is stricter with the combination so that it provides more precision. But WINDOW is kind of a subtle NEAR which does not restrict the order but still have the limit of the distance. In this collection, the sequence may be not that important. What is more important, the distance we used for NEAR is 1 and for WINDOW is 8. These are actually very strict limitation. From the HW1, I only applied NEAR for the special multi-word phrases. As a result, using NEAR for sequential dependency of a query is very strict. This also explain why it is not better than the default combination.