Your Name: Hailun Zhu

Your Andrew ID: hailunz

Homework 3

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 3

1 Experiment 1: Baselines

	Ranked Boolean	**BM25 BOW**	**Indri BOW**
P@10	0.1500	0.3000	0.2300
P@20	0.1800	0.2950	0.2800
P@30	0.1667	0.2967	0.2900
MAP	0.0566	0.1304	0.1277
Time	00:15.5	00:16.78	00:16.89

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

BM25： BM25:k_1= 1.2, BM25:b= 0.75, BM25:k_3= 0

Indri: Indri:mu= 2500, Indri:lambda=0.4

2 Experiment 2: Different representations

	Indri BOW (body)	**0.10 url** **0.10 keywords** **0.10 title** **0.10 body** **0.10 inlink**	**0.70 url** **0.60 keywords** **0.50 title** **0.80 body** **0.40 inlink**	**0.40 url** **0.30 keywords** **0.20 title** **0.80 body** **0.10 inlink**	**0.40 url** **0.30 keywords** **0.20 title** **0.90 body** **0.10 inlink**	**0.04 url** **0.02 keywords** **0.05 title** **0.90 body** **0.01 inlink**
P@10	0.2300	0.1000	0.1500	0.2000	0.2000	0.2300
P@20	0.2800	0.1700	0.1850	0.2500	0.2600	0.2800
P@30	0.2900	0.1733	0.1867	0.2533	0.2567	0.2867
MAP	0.1277	0.0873	0.0909	0.1030	0.1066	0.1246

Describe your strategy for setting the weights on the different representations.

Discuss any trends that you observe; whether the different representations behaved as you expected; the Precision and Recall characteristics of each representation; how the differences in accuracy (if any) relate to different computational cost; and any other observations that you may have.

The strategy I use is to first assign same weight to each field, and then test their influences on performance individually.

I first use the same weight for all the fields, that is to assign no importance to any field. The performance of precision and recall is much worse than the baseline's, so it seems not worth the increased computation cost due to the decrease of the performance. And it is not good to assign uniform weight to every field.

Then I tested the one with 0.04, 0.02, 0.05, 0.9, 0.01. In this case, it is much closer to the baseline's parameters as the body's score is dominant. The performance is very similar as the baseline's performance, but still the performance is worse than the baseline.

I have tested queries with only one field have non-zero weight, the performance of those fields are: body> url> keywords>title>inlink. So I tested queries with 0.7 0.6 0.5 0.8 0.4. The performance has improved compared to the first one which has uniform weight. Both the P@n and MAP value have increased. As I increased the body's weight, both the precision and recall are increased. And as I keep increasing the body field's proportion, the performance keeps improving. This maybe because the body field has more terms so that contains the majority of information.

Then I fix the weight for body is 0.9 and test the other 4 fields by fixing the weight of the 3 of them to be 0.1 and the other one 0.3. The result shows that the performance trend is still: url> keywords>title>inlink. But as different queries may have different preference for these fields and I ignored this by using same structure for all the queries, I could not tell that using url is better than keywords.

The additional computational cost is cost by computing different weights for different fields, calculating and combining those scores.

To sum up, assigning more importance to body field could yield better precision and recall performance. But none of them could exceed the result of only using body field in my experiment. And considering the computational cost has increased using this structure, it may not be a good idea to use this structure when dealing a set of queries.

3 Experiment 3: Sequential dependency models

Example Query: Provide your structured query for query "fickle creek farm".

For weight 0.1 0.1 0.3:

102:#wand(0.1 #and(fickle creek farm) 0.1 #and(#near/1(creek farm) #near/1(fickle creek)) 0.3 #and(#window/8(creek farm) #window/8(fickle creek)))

	Indri BOW (body)	**0.50 AND 0.50 NEAR 0.50 WINDOW**	**0.50 AND 0.25 NEAR 0.25 WINDOW**	**0.25 AND 0.50 NEAR 0.50 WINDOW**	**0.10 AND 0.30 NEAR 0.10 WINDOW**	**0.10 AND 0.10 NEAR 0.30 WINDOW**
P@10	0.2300	0.3500	0.3300	0.3600	0.3500	0.3500
P@20	0.2800	0.3650	0.3500	0.3800	0.3750	0.3650
P@30	0.2900	0.3600	0.3633	0.3567	0.3733	0.3700
MAP	0.1277	0.1892	0.1798	0.1885	0.1856	0.1841

Describe how you set the weights for the different components of the sequential dependency model.

Discuss any trends that you observe; whether the more complex query behaved as you expected; whether the improvement in accuracy (if any) is worth the increased computational cost; and any other observations that you may have.

The strategy I use is to assign same weight to each component, and then test each component's influence on performance individually.

I first assign the same weight to each component. The P@n and MAP results are both better than the baseline's performance, so that by using SDM the precision and recall have both increased. And each component has the same contributions to the final scores.

I then tested the structure that the "and" part dominants. The performance decreased for P@n and MAP, which means the precision and recall are both impaired.

Following this case, I tested the cases in which NEAR and WINDOW are more important. However, only P@n increased and the MAP kept falling when increasing their weights. This means that by assigning more importance on NEAR and WINDOW the precision will increase whereas recall will decrease. This makes sense. By using NEAR and WINDOW, we put more constraints on queries, so that instead of retrieving more relevant documents, we may just retrieve those that are most relevant to the query or most partly relevant to the query, so that increase the precision and decrease the recall.

The additional computational cost is cost by using near and window to get new inverted list and combining the scores of these three parts.

In order to balance the precision and recall, for this set of queries, it is better to assign same weight to these three components. As the overall performance has been improved, I think the improvement in accuracy is worth the increased computational cost.

4 Experiment 4: Multiple representations + SDMs

Example Query: Provide your structured query for query "fickle creek farm".

102: #wand(1 #and(#wsum (0.04 fickle.url 0.02 fickle.keywords 0.05 fickle.title 0.9 fickle.body 0.01 fickle.inlink) #wsum (0.04 creek.url 0.02 creek.keywords 0.05 creek.title 0.9 creek.body 0.01 creek.inlink) #wsum (0.04 farm.url 0.02 farm.keywords 0.05 farm.title 0.9 farm.body 0.01 farm.inlink)) 0 #wand(0.5 #and(fickle creek farm) 0.5 #and(#near/1(creek farm) #near/1(fickle creek)) 0.5 #and(#window/8(creek farm) #window/8(fickle creek))))

	Indri BOW (body)	**w=1.0 (Exp 2)**	**w_1= 0.5**	**w_2= 0.1**	**w_3= 0.2**	**w_4= 0.7**	**w_5= 0.9**	**w=0.0 (Exp 3)**
P@10	0.2300	0.2300	0.2900	0.3500	0.3400	0.2700	0.2700	0.3500
P@20	0.2800	0.2800	0.3450	0.3600	0.3500	0.3350	0.3100	0.3650
P@30	0.2900	0.2867	0.3500	0.3500	0.3600	0.3433	0.3233	0.3600
MAP	0.1277	0.1246	0.1617	0.1853	0.1804	0.1490	0.1334	0.1892

Discuss any trends that you observe; whether the more complex query behaved as you expected; whether the improvement in accuracy (if any) is worth the increased computational cost; and any other observations that you may have.

The improvement by using SDM is significant. So that when weight is 0, that is to only use SDM, I get the best result. And because the performance of using multiple representations is a little bit worse than the BOW's performance, increasing the weight of multiple representations only decrease the precision and MAP values in this case.

Theoretically, the multiple representations model offers a way to provide different representations of the same information by using different fields, thus gives more chances for a query to match the document. This should work well for some kinds of documents. However this may not work well for some other kinds of documents, and this maybe the reason why in my experiments I could not yield good results from multiple representations. In experiment 2, only using body got the best performance, this may imply that it is not very useful to use field-specific tuning and the body field contains the most important information for this set of queries.

The SDM works really well for this set of queries. And the more weight is given to it, the better performance I could get. SDM is often used to convert unstructured queries to structured queries. By implementing SDM, we could get a combine score by combining original score, matching n-grams score, and matching a window constraints' score. By adding additional constraints and structures to the query, we could get a better precision and recall performance.

In conclusion, for this set of queries, it is worthy to use SDM to improve the retrieval performance though the computing cost has increased, whereas using multiple representations may no be so worthy.