Your Name: Hailun Zhu

Your Andrew ID: hailunz

Homework 5

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 5

1 Experiment: Baselines

Provide information about the effectiveness of your system in three baseline configurations.

	BM25	Indri BOW	Indri SDM
P@10	0.4080	0.3800	0.4320
P@20	0.4040	0.3620	0.4220
P@30	0.4013	0.3400	0.4147
MAP	0.2286	0.1994	0.2288

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

BM25:k_1= 1.2

BM25:b= 0.75

BM25:k_3= 0

Indri:mu=500

Indri:lambda=0.1

SDM:0.5 0.5 0.5

2 Custom Features

Describe each of your custom features, including what information it uses and its computational complexity. Explain the intuitions behind your choices. This does not need to be a lengthy discussion, but you need to convince us that your features are reasonable hypotheses about what improves search accuracy, and not too computationally expensive to be practical.

17: Tf*Idf (sum) for body field.

For idf, I use $idf = \log\left(\frac{N+1}{df}\right)$. Equation:

$$Score = \sum tf \cdot \log\left(\frac{N+1}{df}\right).$$

This feature could reflect how important a word is to a document in the corpus. The effect is balanced by the df, so that to avoid favoring documents that have longer lengths. And this is very simple, the time complexity is O(n), where n is the number of unique stems in the documents.

18: Vector Space Model similarity for title field.

The equation is:

$$\sum \frac{d_i \cdot q_i}{\sqrt{\sum d_i^2} \cdot \sqrt{q_i^2}} = \frac{\sum(log(tf)+1) \cdot ((log(qtf)+1)log(\frac{N}{df}))}{\sqrt{\sum(log(tf)+1)^2} \cdot \sqrt{\sum((log(qtf)+1)log(\frac{N}{df}))^2}}$$

I use tf, corpus size and df. I use cosine length normalization. This will test how similar the title is with the original query. I assume the title should contain the important information about the document, so I choose to use to measure the similarity between title and the query. And this is simple. The time complexity is O(n), where n is the number of unique stems in the document, which is the length of the stems in the term vector.

The experiments show that these two features could improve the retrieval performances.

3 Experiment: Learning to Rank

Use your learning-to-rank software to train four models that use different groups of features.

	IR Fusion	**Content-Based**	**Base**	**All**
P@10	0.4640	0.4600	0.4800	0.4760
P@20	0.4140	0.4240	0.4640	0.4620
P@30	0.4120	0.4187	0.4520	0.4533
MAP	0.2493	0.2513	0.2583	0.2584

Discuss the trends that you observe; whether the learned retrieval models behaved as you expected; how the learned retrieval models compare to the baseline methods; and any other observations that you may have.

By adding the IR Fusion (BM25 and Indri scores), both the P@n and MAP values have increased dramatically. As more features are added to the feature vector, the performance has been improved slightly. This behaved as I expected.

Compared to the baseline methods (experiment 1), the P@n and MAP values of LeToR results are all better. But after I put my custom feature into the feature vector, the P@n values have slightly decreased, which may imply that the features I chosen are not so well to represent the information, or it may add some extra not so relevant information into the model.

As for other observations, I have examined the training model and the performance, it shows that for body and title field, the BM25 and Indri features are positive in the model, whereas that of the inlink field are negative. This implies that the scores of different fields have different effect on the learning model, so that influence the retrieval performance in a different way. Also though the term overlap feature is very simple, it still has positive effect on the learning and retrieval performance. This makes sense. Most popular features are not so complex. They help yield a good performance and the computational costs are reasonable. And features like PageRank, which is query-independent could also positively affect the learning model so that to enhance the retrieval performance. This makes sense due to that besides the query-dependent information these features add the document-dependent information into this model, which will influence the learning process.

Also, discuss the effectiveness of your custom features. This should be a separate discussion, and it should be more insightful than "They improved P@10 by 5%". Discuss the effect on your retrieval experiments, and if there is variation in the metrics that are affected (e.g., P@k, MAP), how those variations compared to your expectations.

By adding my custom features, the MAP value and P@30 have slightly increased. This indicates that my features improve the recall. However as the P@10 and P@20 have both decreased a little bit, this means that the precision of the top 20 documents is decreased. This could make sense. The custom features that I added provide more information, including both the relevant information and the redundant information. If this is the case, the recall could be improved and the precision could be decreased. For both f17 and f18, I use tf, df, N. These variables are also used in BM25. So maybe the information they provide has some overlaps. But the following experiments show that in some feature combinations, my custom features could be more useful, which I will explain later.

4 Experiment: Features

Experiment with four different combinations of features.

	All (Baseline)	**$Comb_1$**	**$Comb_2$**	**$Comb_3$**	**$Comb_4$**
P@10	0.4760	0.4880	0.4760	0.4880	0.5240
P@20	0.4620	0.4680	0.4540	0.4640	0.4820
P@30	0.4533	0.4493	0.4467	0.4507	0.4600
MAP	0.2584	0.2587	0.2596	0.2626	0.2720

Describe each of your feature combinations, including its computational complexity. Explain the intuitions behind your choices. This does not need to be a lengthy discussion, but you need to convince us that your combinations are investigating interesting hypotheses about what delivers good search accuracy. Were you able to get good effectiveness from a smaller set of features, or is the best result obtained by using all of the features? Why?

Comb1: disable f18, the computational complexity is almost the same with the baseline. Depending on the program implementation, the complexity could be slightly reduced because we could reduce one iteration through the term vector's stems.

As for intuition, considering the results of the above experiments, my custom features are not so effective as I expected. Then I actually did two tests, one to disable f17, one to disable f18. By disabling f17, only P@10 has been improved and all the other values have been decreased. However, by disabling f18, only P@30 has been decreased whereas all others have been improved. This result indicates that f17 truly has some positive effect on the feature vector as well as the retrieval performance. The information that f17 and f18 used are similar.

Comb2: disable f6, f9, f12, f15, disable all Indri scores. The computational complexity could be slightly reduced because I could reduce several iterations through the term vectors.

As for intuition, by examining the experiment 1 result, I find out that the Indri performance on the bag of words are the worst among the three. This may imply that the Indri does not do very well on this kind of queries, so that I tried to disable all the Indri scores.

The result for disable only all the Indri scores shows that only MAP value has been improved, all the P@n values have been slightly impaired. This does not meet my expectation, according to the Indri Bow scores, all P@n and MAP values are worst. As for my understanding, applying Indri on structured queries (like SDM queries) could yield better performance, but applying Indri on unstructured queries maybe worse.

Another finding in this combination is that I have tested two sets, one with Indri disabled, and one with Indri and f18 disabled. In contrast to combination 1, using f18 could better the performance with Indri scores disabled. This may imply the effect of each feature is not simply "added" to the overall effect, and the features are correlated in someway and the combinations of the features really matter to the final performance.

Comb3: disable f6, f7, f9, f10, f12, f13, f15, disable all Indri scores and some term overlap scores. The computational complexity could be slightly reduced because I could reduce several iterations through the term vectors.

This combination disables some term overlap scores on the basis of comb2, the reason why I do this is that considering term overlap is a very simple feature, it may add more redundant information than useful information, so that this could worsen the retrieval performance. Also because term overlap score does not take document length into account, in other words there is no idf like effect in this feature, this feature may favor longer documents (longer documents are more likely to have the query term).

The result meets my expectation. Compared to the baseline, except for P@30, all the other values have been improved, which means both the recall and precision are both improved. Considering the same method applied to different fields could yield different effects, I also have tested two sets, one with all term overlap scores disabled (f7, f10, f13, f16), the other one with some term overlap scores disabled (f7, f10, f13). The result shows that enabling term overlap scores in inlink field could slightly improve the

result. This may be because the inlink field includes important information about the content, and it is often not too long.

Comb4: disable f6, f7, f9, f10, f11, f12, f13, f15, f17. The computational complexity becomes lower due to fewer features are needed to be computed.

For this combination, I firstly disabled the first 4 query-independent features individually. None of the results shows any improvement. This indicates that the document-dependent features are important in this model. These features could complement the model by adding document information. Considering this, I enable all the first four features. Then as I have stated before, same method used in different fields may yield different results, and I recall in HW1, the url field does not contribute much to the result, so I also disabled f11 feature, which is BM25 in url field. The retrieval performance has been improved. And considering that f17 feature could be considered as a simplified version of BM25, I tried to disable it and also got some improvement. This maybe because that by using BM25 for body field is enough for collecting information of tf and idf. Thus, introducing more features could adding more redundant information than useful information and therefore impairs the results.

I am able to get good effectiveness from a smaller set of features. I think the major reason is that the information the other disabled features provided introduces more irrelevant information, thus expends the search range and declines the recall and precision values. Also, some features are correlated to others. There are some overlaps of the information they provided. Hence, there is no need to include both the features. What's more, through all these experiments, I realize that simple features do not mean worse performance. Simple features could also contribute a lot to the feature vector, thus enhance the retrieval performance. And the query-independent features are essential to the feature vector.

5 Analysis

Examine the model files produced by SVM^{rank}. Discuss which features appear to be more useful and which features appear to be less useful. Support your observations with evidence from your experiments. Keep in mind that some of the features are highly correlated, which may affect the weights that were learned for those features.

Some of this discussion may overlap with your discussion of your experiments. However, in this section we are primarily interested in what information, if anything, you can get from the SVM^{rank} model files.

Through all the experiments that I did, BM25 for body field always has the highest weight. As the test documents are retrieved from BM25 in body field, this makes sense. And according to the experiment one, for bags of words (unstructured queries), BM25 has better performance than Indri, so that using BM25 is better and this feature appears to be more useful.

The spam score feature usually will have higher weights. The spam score is the most useful feature among the query-independent features. This means the quality of the document is very useful for retrieval performance.

Disabling BM25 in body field and spam score will degrade the recall and precision a lot. This behaves correspond to the model file. As their weights are very high, their values are important for positive classes, removing them will impair the model.

By disabling different features, the weights of other features could change.

Using all the features, it shows that f6, f7 have higher weights, which is the Indri and term overlap ratio for body field.

1 1:0.51557887 2:-0.13147865 3:0.2931329 4:0.016699929 5:0.53853774 6:0.30085999 7:0.30904135 8:0.20013654 9:0.028402571 10:0.31747213 11:0.1256475 12:0.12449641 13:0.24974252 14:-0.0045066774 15:-0.048295308 16:0.073243551 17:0.13041189 18:0.19303365 #

These two features may be correlated with the f5, for they all use body field term vector to get the results, so that their weights are overestimated. The following experiment results show that disabling these features could improve the performance.

By disabling f6, f7, f9, f10, f11, f12, f13, f15, f17, the weights are:

1 1:0.52093983 2:-0.10186484 3:0.35099357 4:0.0043514492 5:0.70688128 8:0.43656278 14:0.080032676 16:0.16005796 18:0.37221357 #

I find out that f14's weight is negative in "all" experiment, but positive in the comb4 experiment. Using all the features introduces extra "noise" into the model, which could make useful features less useful as well as make redundant feature appear to be useful. Though f6, f7, f13 features have high weights, removing them yields better result. This implies that they are less useful even harmful.

I then emphasis on the comb4 experiment, because the comb4 has the best result in my experiments. The model file shows that BM25 for body, spam score, Fromwikipedia score are the most useful features because they have the highest weights. Though the PageRank score's weight is close to zero, removing it may slightly impair the performance. This means it more or less has some positive effect on the retrieval model. But the effect is small. By removing it, other features' weights are improved- positive value got a bit bigger, negative value got a bit smaller.

The f2 has negative value, which is also important to the model. This means that it is correlated with negative class. Removing it also worsen the result.