Information Retrieval 5M - Exercise 2

Mayra A. Valdes Ibarra - 2419105v

1 Deployment and evaluation of a baseline LTR approach

Learning-to-rank is a recent paradigm used by commercial search engines to improve retrieval effectiveness by combining different sources of evidence. The goal of this report is to analyze, test and evaluate learning-to-rank features described in the article `Ronan Cummins and Colm O'Riordan, 2009`[1] to see how pairwise term-term proximity can boost the scores of documents using Terrier's support for learning-to-rank (LTR) and the Jforests LambdaMART LTR technique.

We first analyze the effectiveness performance of two system configurations: `LTR (using PL2 to generate the sample)` vs. `PL2` on `HP04 Topic Set`. Table 1 below shows the results of the analysis for two different metrics, `MAP` and `P@5` (Performance at Rank 5).

Table 1: Performances of LTR vs PL2 on HP04 Topic Set

	Metric	
Configuration	MAP	P@5
PL2	0.2251	0.0693
LTR (PL2 sample)	0.4690	0.1333

We proceed to perform t-test significance tests to conclude if the performance of `LTR` is on average significantly better than `PL2` for any of the metrics. Our null hypothesis is that the mean performance of `LTR` is the same as the mean of `PL2` for both metrics. After running the t-tests, the following results were obtained:

`LRT` vs `PL2` on `Metric MAP`: The t-value is -4.1101. The p-value is 0.000067. The result is **significant** at $p < 0.05$.

`LRT` vs `PL2` on `Metric P@5`: The t-value is -3.4534. The p-value is 0.000737. The result is **significant** at $p < 0.05$.

Since the p-value is < 0.05 for both metrics, we reject the null hypothesis and conclude that there is statistically significant evidence that on average `LTR (PL2 sample)` is better than `PL2` on both, `MAP` and `P@5` metrics.

2 Proximity Search Features

2.1 Choosing which Proximity Features to Implement (Question 2a)

Based on the proposed paper `Ronan Cummins and Colm O'Riordan. 2009`, two proximity features were implemented and analyzed.

[1] Ronan Cummins and Colm O'Riordan. Learning in a pairwise term-term proximity framework for information retrieval. In SIGIR 2009, pages 251–258, Boston, MA, USA, 2009. ACM. https://dl.acm.org/citation.cfm?doid=1571941.1571986

The proximity features that are implemented are `min_dist` *(Minimum Distance Proximity Feature)* and `avg_distance` *(Average Distance Proximity Feature)*. These features are analyzed individually and in combination on `HP04 Topic Set`.

Minimum Distance Proximity Feature. One of the reasons for choosing `min_dist` is that it appears to be very efficient even though it requires less computation complexity than the rest of the possible features to implement. With the implementation of this feature, we expect to see a boost on documents where two of the query terms appear next to each other (or very close).

Average Distance Proximity Feature. After implementing this feature, we expect to see a boost in documents where query terms appear in approximately the same locations in the document. It is expected to see a penalty (decrease on performance) on documents where one of the query terms appears further away than other terms, as it will increase the average distance.

2.2 Implementation of the Proximity Search Features (Question 2b)

Source code is provided with the implementation for both features. It was quite of a challenge to decide what was the best approach to implement the features as well as the aggregating function for both features. The following assumptions were made when implementing the features:

- The weights for each term is not taken in mind
- The absence of key terms in a document is not penalized
- We iterate through all possible combinations of the terms in the query, which in theory is Full Dependency (compared to Sequence Dependency used for baseline)

The biggest challenge faced was the absense of a query term. We made sure to make proper use of `okToUse` by implementing some unit tests with different values. We think that the absense of a term is key, and should be further evaluated and considered.

Minimum Distance Implementation. It was decided that the best possible aggregating function was to get the $min(min\_dist_{i,j \in D \cap Q, \forall i \neq j}(i,j))$, where i and j denote the i^{th} and j^{th} term respectively, D denotes Document and Q denotes Query. The reason for this choice is that the goal is to boost documents where the distance between query terms is small.

Average Distance Implementation. After testing different algorithms, it was decided that the best possible aggregating function for this feature was to get the $sum(avg\_dist_{i,j \in D \cap Q, i<j}(i,j))$. Other aggregating functions were tested, but it was concluded that `sum` had the best performance.

One of the difficulties faced when implementing this features was to make sure that the algorithm detects the right score in any order and for all combinations of each query term. The unit tests reflect different possible combinations on queries with `n` terms and different distances.

2.3 Testing of the Implementation of Proximity Search Features (Question 2c)

Unit tests were implemented for different combinations of documents and query terms. Queries were tested for 2 or more terms. Different terms combinations were tested. For example for query "a b", documents with terms "a" term before "b", "b" before "a", "a" missing terms in document, as well as other combinations were taking in mind. One of the special thigns to test was the complexity computation by not doing extra loops in the `min_dist` feature, where many errors were found and fixed when implementing the algorithm.

3 Evaluation of the Proximity Search Features

We proceed now to evaluate the proximity features implemented in Terrier. The implementations are called `Dependency Score Modifiers (DSM)`. Let `DSM1` denote the configuration of first feature implemented, in this case $min\_dist(a, b, D)$, let `DSM2` denote the second feature implemented, in this case $avg\_dist(a, b, D)$. These features were evaluated individually and in combination along with `LTR baseline` on HP04 Topic Set. Table below shows the results of the analysis.

Table 2: Performances of DSM1, DSM2 with LTR baseline on HP04 Topic Set

	Metric	
Configuration	MAP	P@5
LTR (baseline)	0.4690	0.1333
LTR + DSM1	0.5032	0.1413
LTR + DSM2	0.4907	0.1360
LTR + DSM1 + DSM2	0.5356	0.1360

`LRT (baseline)` vs `LTR + DSM1` on Metric MAP: The t-value is -0.5147. The p-value is 0.6075. The result is **not significant** at $p < 0.05$.

`LRT (baseline)` vs `LTR + DSM1` on Metric MAP: The t-value is -0.3243. The p-value is 0.7462. The result is **not significant** at $p < 0.05$.

`LRT (baseline)` vs `LTR + DSM1 + DSM2` on Metric MAP: The t-value is -0.9775. The p-value is 0.3299. The result is **not significant** at $p < 0.05$.

We see an improvement on all combinations for both metrics with respect to the baseline. However, the improvement is not statistically significant.

4 Performance of the learned model

Using MAP as the main evaluation metric, below we provide a summary of the performance of the learned model with and without the additional proximity features.

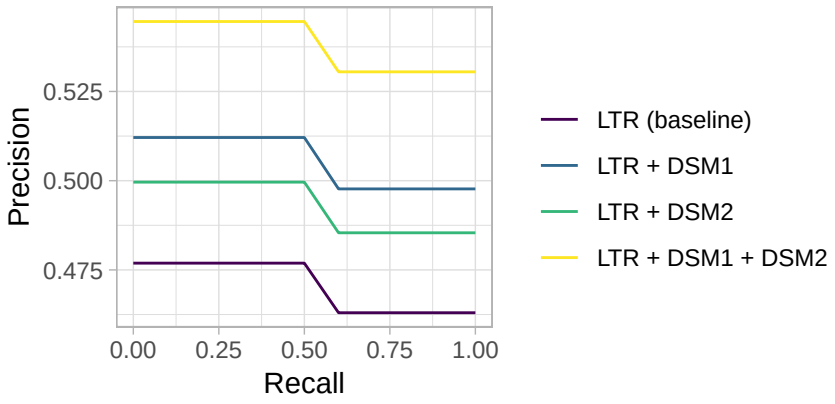

Figure 1: Recall-Precision Plot of the 4 system variants used

Figure 1 above shows a Recall-Precision graph where we can observe how any combination of proximity feature (either individually or in combination) outperforms the LTR baseline. In particular the combination `LTR + DSM1 + DSM2` outperforms `LTR (baseline)` by 0.07.

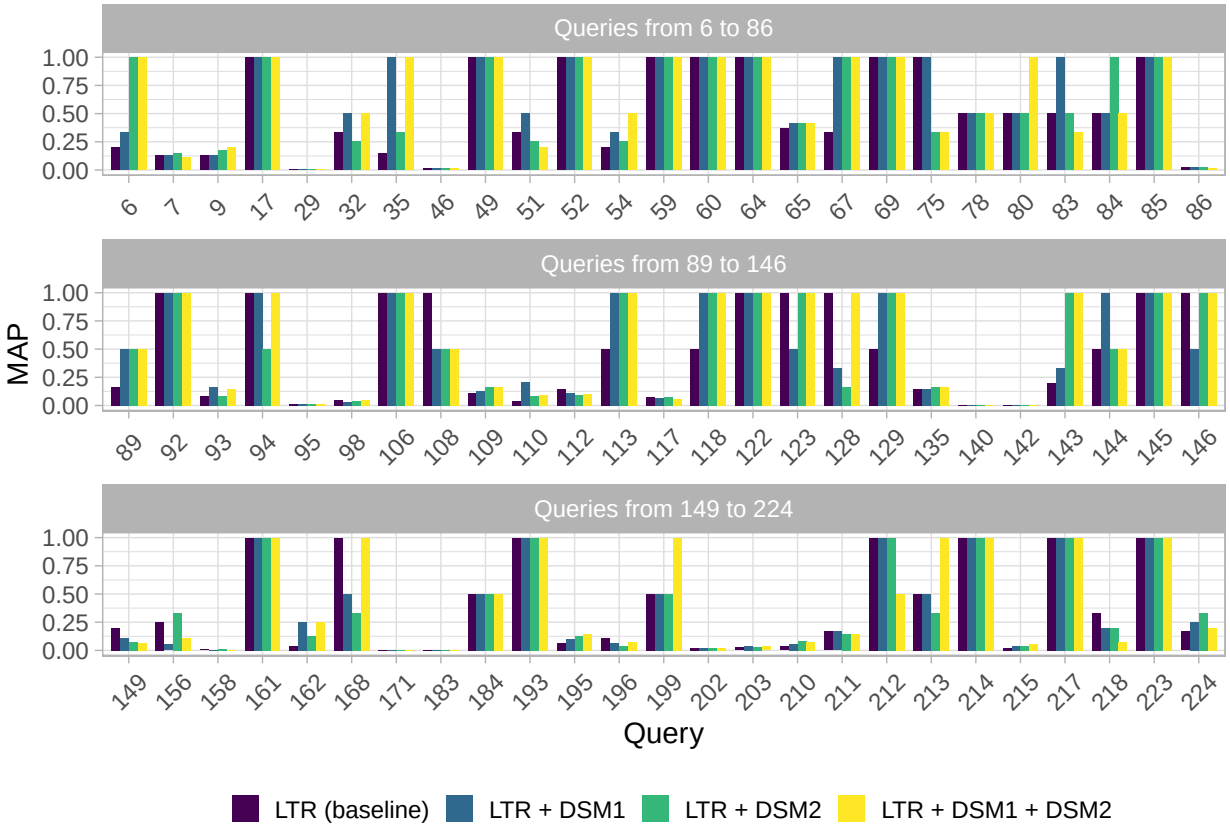

Figure 2: Histogram with a query-by-query performance analysis of the 4 system variants used.

Figure 2 above shows a histogram of query-by-query performance analysis of the 4 system variants used. It is interesting to see how in different queries the combination of both proximity features improves the performance of the query, but in some other queries it does not. It seems that this does not depend to whether each proximity feature improved the performance by itself.

We now proceed to summarize the amount of the number of queries that have been improved, degraded or remained unaffected with the introduction of any variant of the implemented features with respect to the `LTR baseline`. Table below shows a summary of all variants and their respective performance scores. It is important to mention that even though the variant of `LTR + DSM1 + DMS2` had the best `MAP score` (0.5356), as mentioned in Table 2, we can also see in Table 3 that it also degraded a higher amount of queries than `LTR + DSM1` and `LTR + DSM1` variants.

Table 3: Performance summary of the implemented proximity features

Configuration	Total	Improved	Unaffected	Degraded
LTR + DSM1	75	25	35	15
LTR + DSM2	75	26	32	17
LTR + DSM1 + DSM2	75	26	31	18

Tables below show examples of queries that have been particularly improved or harmed with the introduction of the implemented proximity search features.

Table 4: Summary of query terms that improved/degraded in MAP score by 0.5 or more.

		MAP Score			
#	Query Terms	Baseline	DSM1	DSM2	DSM1 + DSM2
6	Philadelphia streets	0.2000	+ 0.1333	+ 0.8000	+ 0.8000
35	Religious Freedom amendment	0.1429	+ 0.8571	+ 0.1904	+ 0.8571
67	State local gateway	0.3333	+ 0.6667	+ 0.6667	+ 0.6667
75	NOAA Fisheries Northwest Region	1.0000		- 0.6667	- 0.6667
83	NSA Home for kids	0.5000	+ 0.5000		- 0.1667
84	National Institute on Alcohol Abuse and Alcoholism homepage	0.5000		+ 0.5000	
94	Florida Keys Marine sanctuary	1.0000		- 0.5000	
108	Panic disorder NIMH	1.0000	- 0.5000	- 0.5000	- 0.5000
113	Fermilab Flora Fauna	0.5000	+ 0.5000	+ 0.5000	+ 0.5000
118	California board of corrections	0.5000	+ 0.5000	+ 0.5000	+ 0.5000
123	Cave exploring lesson plan	1.0000	- 0.5000		
128	Planetary balloon program	1.0000	- 0.6667	- 0.8333	
129	NODC coastal water temp	0.5000	+ 0.5000	+ 0.5000	+ 0.5000
143	FEMA for kids hurricane facts	0.2000	+ 0.1333	+ 0.8000	+ 0.8000
144	California Tahoe conservancy	0.5000	+ 0.5000		
146	astrobiology	1.0000	- 0.5000		
168	Maryland Rural development	1.0000	- 0.5000	- 0.6667	

We can see in table above how some terms improved individually, while others degraded individually but were not degraded while using a combination of features. Further evaluation needs to be performed to see if there is any term missing that could have penalized the relevant documents. We can see that query 128 *(Planetary balloon program)* is one of the queries that got the highest penalty with both features when they were evaluated individually, but it was unnafected when evaluating it with both features combined. Queries like *FEMA for kids hurricane facts*, *NODC coastal water temp* and *Fermilab Flora Fauna* were improved by both features individually and in combinations, which could mean that queries with uncommon words are improved.

5 Conclusions

Given the results, we can conclude that even though we did not achieve statistically significant improvements in MAP performance after the implementation of proximity features with respect to the LTR baseline, we can still see that they do help to boost relevant documents.

Further improvements could be done to the algorithms to the implementation by taking in mind the weights of the terms and a possible penalization of absense of query terms in a document, and possibly changing the aggregating function by normalizing the score based on the number query terms used.