Information Retrieval

MOSTAFA KHATTAT, MARKO MATUŠOVIČ, and LIANG ZHANG

ABSTRACT

Information retrieve models essentially aim to get all relevant documents for a query and rank them. In this paper, we research probabilistic model BM25 and 8 popular learning to rank algorithms. We decide to evaluate the 200 test queries and collection of passage ranking datasets. All datasets are from TREC 2019 Deep Learning track[1]. And we perform some error analysis by sampling a few queries that have achieved very low retrieval effectiveness to improve the baselines. The repository that contains the software we used, the scripts we used to analyze our data, etc are accessed on our GitHub page[1].

1 INTRODUCTION

The Deep Learning Track at TREC 2019 [1] looks at ad hoc ranking in a big data environment. It is the first track to include large human-labelled training sets, with two sets matching to two tasks (document and passage ranking), each with TREC-style reusable test sets. The corpus for document retrieval is 3.2 million documents with 367 thousand training queries, whereas the corpus for passage retrieval is 8.8 million passages with 503 thousand training queries. Both tasks consist of a 43-question reusable exam set. The raw data of documents and passage ranking dataset are 22GB and 2.9GB respectively. We decide to build both baselines on the passage ranking dataset.

TREC made queries, collection of documents and human judgement files public. This allows any researchers to perform retrieval models without limiting by datasets, especially machine learning-based models which require very large datasets. Besides, TREC report[2] gives some performance indicators of the state of the art methods.

Our goals for this project:

- build two baselines(one for probabilistic model and one for learning to rank)
- propose improvements on at least one baselines
- share our findings and challenges during the project

2 APPROACH

We make a step-by-step plan to examine and improve retrieve results:

(1) get familiar with TREC 2019 deep learning by understanding the format of collection and query of passage ranking dataset and make a selection what to use for two baselines.
(2) select a probabilistic model and learning to rank algorithm.
(3) use combination of available software and custom developed code to replicate our selected model.
(4) Perform error analysis on results(e.g. check the documents not retrieved but relevant).
(5) suggest improvements to the baseline by incorporating new techniques.

[1] https://github.com/lzhanggithub/IR-Group-42

Authors' address: Mostafa Khattat, m.khattat@student.tudelft.nl; Marko Matušovič, m.matusovic@student.tudelft.nl; liang Zhang, l.zhang-31@student.tudelft.nl.

2.1 BM25

For the probabilistic model, we focused on the BM25 model. It is an effective and robust model since it includes many criteria including term frequency, document weight and query term frequency [3]. To perform indexing and retrieving of the relevant documents we used Anserini [2] toolkit which is built on top of Apache Lucene. For our task collection we chose the passage ranking dataset which contains 8, 841, 823 records and for our test collection we used MS-Marco 2019 with 200 test queries [1].

2.2 Learning to Rank Algorithms

We selected passage ranking dataset to perform learning to rank model baseline, since its file size is smaller. In this task, we used qrels.train.tsv, qrels.dev.tsv, and 2019qrels-pass.txt [1] as our training, validation and test data respectively. We used toolkit RankLib[4] to perform eight popular algorithms: MART (Multiple Additive Regression Trees, a.k.a. Gradient boosted regression tree) [5] RankNet [6] RankBoost [7] AdaRank [8] Coordinate Ascent [9] LambdaMART [10] ListNet [11] Random Forests [12].

The format of training and validation files are TREC format: TOPIC/ITERATION/DOCUMENT/RELEVANCY. **TOPIC** is the query number, **ITERATION** is the feedback iteration (almost always zero and not used), **DOCUMENT** is the official passage id that corresponds to the "docno" field in the passage, **RELEVANCY** is a binary code of 0 for non-relevant and 1 for relevant.

However, there is a challenge for us to prepare the training and validation file. There are only relevant files with only relevance label 1. The same target label (i.e. 1) for all passages is not feasible for learning to rank algorithm. In order to get some non-relevant documents we first ran the Anserini to retrieve the top 10 documents for each query and then we compared the result with the relevance judgement file. For each query, the retrieved documents that were in the relevance file received a label 1 (relevant) and other documents received a label 0 (non-relevant) in the input file for the learning to rank algorithm. For the test file, we used 2019qrels-pass file which already contains a score for each relevant document.

3 EXPERIMENTS

In this section, we evaluate our models using NDCG@10 and MAP@10, since They are the most widely used offline evaluation metrics today. Besides, we can compare the NDCG@10 score of our model with the TREC report. NDCG is useful because it helps measure the usefulness or gain of a document based on its position and MAP fits our evaluation since it takes the order in which documents are returned into account. Furthermore, these two metrics are available for both BM25 and learning to rank so we can get an idea of how these two models performed. [2].

3.1 BM25

Initially, we indexed the whole main collection and ran the Anserini tool to retrieve the relevant documents based on the msmarco-test2019 queries. On a machine with four cores and an NVME drive, it took about two minutes to build the index. For BM25 we chose the parameters $k1 = 0.6$ and $b = 0.9$. We used a tuning tool provided by Anserini which

[2] https://github.com/castorini/anserini

	Metric	Result
BM2	Normalized Discounted Cumulative Gain (NDCG)@10	0.2253
	Mean Average Precision (MAP)@10	0.1115
LambdaMART	NDCG@10	0.3903
	MAP@10	0.619
MART	NDCG@10	0.3611
	MAP@10	0.5857
RankNet	NDCG@10	0.1791
	MAP@10	0.4336
RankBoost	NDCG@10	0.2647
	MAP@10	0.4935
AdaRank	NDCG@10	0.2631
	MAP@10	0.483
Coordinate Ascent	NDCG@10	0.431
	MAP@10	0.6423
ListNet	NDCG@10	0.4268
	MAP@10	0.6345
Random Forests	NDCG@10	0.3848
	MAP@10	0.5593

Table 1. matrix results of NDCG@10 and MAP@10 on BM25 baseline and on 8 learning to rank algorithms

Feature	unique terms in document	document length	term frequency	inverse document frequency	BM25 score
NDCG@10	0.0952	0.2465	0.2131	0.1793	0.4153
MAP@10	0.3569	0.4587	0.4824	0.3756	0.6283

Table 2. NDCG@10 and MAP@10 of each feature used to training ListNet

uses a grid search approach of parameter values in tenth increments to find the best values for our queries. Table 1 shows the initial results for baseline BM25.

3.2 Learning to Rank Algorithms

We compute five features to train all eight algorithms: unique terms in documents (document feature), document length (document feature), term frequency, inverse document frequency and BM25 scores. The challenge we overcome here is that we have to write python scripts(see Github repository) to compute each score, while some software can automatically calculate feature scores and use them to train models(e.g. pyterrier [3]). With the results of Table 1, we know ListNet is the best algorithm for this passage ranking dataset. Hence, for this algorithm, we also compute the metric of each feature which is shown in Table 2.

Since ListNet has the best performance as in Table 1 (highest scores in both NDCG@10 and MAP@10), we decided to further examine the effectiveness of 5 features on ListNet which are presented in Table 2 and shows that the most effective features is BM25. The reason is that BM25 consists of term frequency, length of the document, average document length, and inverse document frequency.

[3] https://github.com/terrier-org/pyterrier

By comparing Table 1 to Table 2, we notice that NDCG@10 of ListNet (0.4268) and MAP@10 (0.6345) of ListNet trained by 5 features are about 1% higher than that trained only by feature BM25 score(NDCG@=0.4153, MAP@10 = 0.6283). Since the BM25 includes all features except unique terms in the document, we wonder if the combination of features (unique terms in documents and BM25) can train ListNet with a better performance. Hence we trained ListNet with two features (unique terms in documents and BM25), and we got NDCG@10 = 0.4083, and MAP@10 = 0.6248. This result means the combination of two features fails to give a better performance for ListNet. Therefore, in future work, we can train models with more features and test combinations of them to get the best performance of models.

4 ERROR ANALYSIS

With the results of experiments we collected a list of best-ranking documents for each query, however, our baselines of BM25 (NDCG@10 = 0.2253) show lower performance compared to the state of the art methods (NDCG@10 >= 0.4495). Sometimes relevant documents are not retrieved and instead non-relevant documents are retrieved. This section describes the process of error analysis of the baseline.

First, the outline of the special cases to be analysed was created. The cases that were investigated we inspired by the work of Buckley and Harman [13]. Namely:

- Documents that are not related to query but retrieved by the model and placed among top positions.
- Documents that are related to query in relevance judgement file but not retrieved.

A python script was developed to process the results of the experiment and match the ids with the texts of query and passage in the data sets. This code can be found in our GitHub repository. The actual analysis of the queries was performed individually, the results were compared and discussed together. For the error analysis, we looked at the **QUERY:19335**: anthropological definition of environment. The final observations are discussed below per type of error.

Partial Query Match. When a document contains only a subset of terms, that are found in the query, they are often not retrieved. This is a problem since they can still be relevant. Consider the following example:

- **DOC:1720389** (not retrieved) Definition of Environment. Environment: The sum of the total of the elements, factors and conditions in the surroundings which may have an impact on the development, action or survival of an organism or group of organisms. ...
- **DOC:2046505** (not retrieved) Definitions of Anthropological Terms. academic anthropology - careers that involve the teaching of anthropology at colleges and universities. Academic anthropologists do research, but the objective is more for the contribution to general knowledge. ...

The query asks for a definition of environment, the two documents are relevant however they were not retrieved. The documents only contain a subset of terms. There is no term "*anthropological*" in **DOC:1720389** and there the term "*environment*" is not present in **DOC:2046505**. Because the documents are short, matching a term is relatively important. In a case one term is missing completely, the document is not retrieved.

Lexical Category Variation. Terms can have various lexical categories and still have the same, or similar, relevance. Since all query and document features used in our baselines required an exact match of terms, in a case the word has a different form or is changed even a little, the term does not match and the document is ranked lower. For our example query 19335, there are twenty relevant documents from which seven are highly relevant. The baseline model BM25

could only manage to get three of them at the top ten and six of them at the top twenty. Below are some examples of highly relevant documents that were not among the top ten results:

- **DOC:8412683** (ranking: 30) Ecological anthropology is defined as the study of cultural adaptations to environments. The sub-field is also defined as, the study of relationships between a population of humans and their biophysical environment. …
- **DOC:8412682** (ranking: 15) Environmental anthropology is a sub-speciality within the field of anthropology that takes an active role in examining the relationships between humans and their environment across space and time …
- **DOC:8412681** (ranking: 14) From Wikipedia, the free encyclopedia. Ecological anthropology is a sub-field of anthropology and is defined as the study of cultural adaptations to environments …

We suspect that this issue is due to the fact that the model uses exact term match between queries and documents instead of matching documents at a semantic level to perform the retrieval. One solution would be to use document expansion to decrease the semantic gap. This method will be covered in the next section.

5 IMPROVEMENTS

To reduce the semantic gap in the retrieved result by the search engine, many approaches can be employed. One of these approaches is document expansion. The basic idea is to train a model that given an input document, it generates queries for which the document might be relevant. These predicted queries are then appended to the original documents which are then indexed [14]. We used msmarco-passage-pred-test_topk10 dataset [4] to expand our document collection and re-index it.

For the lexical category variation error analysis, the baseline could only manage to get three out of seven highly relevant documents at the top ten whereas this improved version could manage to get five out of seven. Table 3 shows the overall result over the whole collection. As it can be seen, document expansion did slightly improve the baseline model for our query test cases in the msmarco-test2019 file(NDCG@10 from 0.2253 to 0.2328, MAP@10 from 0.1115 to 0.1144).

Another improvement that was implemented is query sanitising. The queries were processed with our code that can be found on our GitHub repository in the file "query_processing.py". Queries are split into terms and all punctuation is removed. Moreover, the stop-words are also removed from queries. Removing stop words from the query allows the model to give higher weight to terms that are more important.

Query sanitising is a simple improvement and it raises the score of MAP@10 from 0.1115 to 0.1129, however the NDCG@10 drops from 0.2253 to 0.2232. This can be explained by the fact that more relevant documents were retrieved, however they were ranked lower.

[4] https://git.uwaterloo.ca/jimmylin/doc2query-data/raw/master/base/msmarco-passage-pred-test_topk10.tar.gz

Method	NDCG@10	MAP@10
BM25	0.2253	0.1115
BM25 with document expansion	0.2328	0.1144
BM25 with query sanitising	0.2232	0.1129

Table 3. The result of improvements proposals

6 CONCLUSIONS

Our baselines are based on BM25 and 8 learning to rank algorithms which are both already robust information retrieval models. Processing and evaluating the data sets of MS MARCO allowed for analysis of the erroneous retrievals. The error analysis revealed two main issues, relevant documents were not retrieved in case the query contained additional terms absent in the document, and if the lexical category of the terms differed. We proposed two improvements to BM25, document expansion and query sanitising. The retrieval models with the improvements implemented produced a higher score on metrics.

REFERENCES

[1] MS MARCO. Trec 2019 deep learning track guidelines. https://microsoft.github.io/msmarco/TREC-Deep-Learning-2019#deep-learning-track-tasks.

[2] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. Overview of the trec 2019 deep learning track. *arXiv preprint arXiv:2003.07820*, 2020.

[3] Stephen Robertson and Hugo Zaragoza. *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc, 2009.

[4] Van Dang and Michael Zarozinski. Ranklib overview.

[5] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[6] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, pages 89–96, 2005.

[7] Yoav Freund, Raj Iyer, Robert E Schapire, and Yoram Singer. An efficient boosting algorithm for combining preferences. *Journal of machine learning research*, 4(Nov):933–969, 2003.

[8] Jun Xu and Hang Li. Adarank: a boosting algorithm for information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 391–398, 2007.

[9] Donald Metzler and W Bruce Croft. Linear feature-based models for information retrieval. *Information Retrieval*, 10(3):257–274, 2007.

[10] Qiang Wu, Christopher JC Burges, Krysta M Svore, and Jianfeng Gao. Adapting boosting for information retrieval measures. *Information Retrieval*, 13(3):254–270, 2010.

[11] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, pages 129–136, 2007.

[12] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[13] Chris Buckley and Donna Harman. Reliable information access final workshop report. 03 2004.

[14] Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*, 2019.