bigIR at TREC 2020: Simple but Deep Retrieval of Passages and Documents

Fatima Haouari, Marwa Essam, Tamer Elsayed
{200159617,me1709534,telsayed}@qu.edu.qa
Computer Science and Engineering Department, Qatar University

ABSTRACT

In this paper, we present the participation of the bigIR team at Qatar University in the TREC Deep Learning 2020 track. We participated in both document and passage retrieval tasks, and each of its subtasks, full ranking and reranking. As it is our first participation in the track, our primary goal is to experiment with the latest approaches and pre-trained models for both tasks. We used Anserini IR toolkit for indexing and retrieval, and experimented with different techniques for passage expansion and reranking, which are either BERT-based or sequence-to-sequence based. All our submitted runs for the passage retrieval task, and most of our submitted runs for the document retrieval task outperformed TREC median submission. We observed that BERT reranker performed slightly better than T5 reranker when expanding passages with sequence-to-sequence based models. However, T5 achieved better results than BERT when passages were expanded with DeepCT, a BERT-based model. Moreover, the results showed that combining the title and the head segment as document representation for reranking yielded significant improvement over each separately.

1 INTRODUCTION

Motivated to study the use of deep learning approaches in ad-hoc search over large-scale datasets, the Text REtrieval Conference (TREC) organized the deep learning track (TREC-DL) in 2019, with a follow-up in 2020. The track employs MS MARCO collection,[1] a large-scale Question Answering (QA) dataset created from around half a million questions sampled from Bing's search query logs. QA is mainly concerned with developing systems that can provide answers to questions posted by humans. The dataset contains 3.2M documents, with around 8.8M passages, and over 1M queries.

The deep learning track has two tasks, document retrieval and passage retrieval. In both tasks, a set of questions are given, and the goal is to find answers to these questions from the MS MARCO dataset. In document retrieval, the task is to retrieve, for each question, a list documents that most probably contain the answer to the given question. While in passage retrieval, the task is to retrieve the actual passages (from within the documents) that most probably contain the answer to that question. The same test queries are used for both the passage retrieval and document retrieval tasks, with NDCG@10 being used as the official measure for evaluation.

The most successful approaches in TREC-DL 2019 adopted passage expansion and exploited BERT for reranking [11, 14]. The top ranked runs for both passage and document retrieval tasks were by Yan et al. [11]. For the passage retrieval task, they trained an encoder-decoder model with their proposed "attention over attention" mechanism to expand the passages. To rerank the retrieved passages and documents, they first trained a BERT model from scratch by modifying the next sentence prediction task. They then fine-tuned it using the query-passage MS MARCO collection by employing the point-wise ranking technique. For the document retrieval task, they first split the document into passages to overcome the input size limitation of BERT. For each document, they reranked the split passages concatenated with the title of documents, and they considered the maximum relevance score among all passages as the document score. The second best performing runs were submitted by Yilmaz et al. [14] who used a pre-trained transformer model [8] to predict queries given passages, and they expanded each passage in the collection with the predicted queries. They employed a BERT-based relevance classifier pre-trained by Nogueira and Cho [5] for reranking passages and documents. For document reranking, they split each document into sentences and they used BERT to rerank those sentences. Finally, to rerank documents, they used an aggregation of the top 3 sentences scores.

In this paper, we describe our participation for both the passage and document retrieval tasks in 2020. We adopted a simple approach that employs document expansion, query expansion, and reranking models. For document expansion, we adopted BERT-based and sequence-to-sequence based approaches. For the passage retrieval task, prior indexing, we expanded each passage in the collection with a set of queries for which the corresponding passage may contain their answers. We experimented with three different passage expansion techniques, namely doc2Query [8], doctTTTTTquery [7], and DeepCT [3]. To further improve the retrieval, we expanded the queries using RM3 [1]. For reranking, we exploited two different pre-trained models, namely monoBERT [5] and monoT5 [6]. To alleviate the input length limitation of the reranking models, we tried different short document representation for the document retrieval task.

The rest of the paper is organized as follows: In Section 2, we detail our participation for the passage retrieval task, including a description of the submitted runs and an analysis of the results. Similarly, in Section 3, we present our participation for the document retrieval task. Finally, we conclude in Section 4.

2 PASSAGE RETRIEVAL TASK

In this Section, we present our approach and experimental results for the passage retrieval task. We present our general approach in Section 2.1. We discuss the preliminary experiments we performed before our official runs configurations selection in Section 2.2, then we present our official submitted runs for both the full ranking and reranking subtasks in Section 2.3. Finally, we discuss the results of our official runs in Section 2.4.

[1] http://www.msmarco.org/

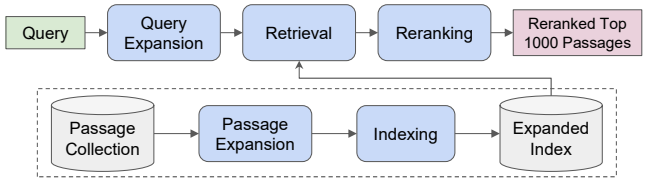

Figure 1: Proposed approach for the passage retrieval task. The process within the dotted rectangle is only done for the full ranking subtask.

2.1 Approach

The passage retrieval task has two subtasks: full ranking and reranking. For the full ranking subtask, we adopted a *two-way expansion approach*. We first *expanded each passage* in the collection with queries for which this passage most probably contains their answers. For that, we used a pre-trained passage expansion model. The expanded passages are then indexed. We further used *query expansion* to expand each query with a set of keywords to further improve the retrieval. The expanded queries are used to retrieve an initial candidate set of passages for each given query. Finally, the candidate set is reranked using a pre-trained reranker model. For the passage reranking subtask, we reranked the top 1000 passages given by TREC organizers. Figure 1 illustrates the approach.

We based all our experiments on passage and query expansion which were showed to improve retrieval performance in the TREC-DL 2019 [2].

For passage expansion, we adopted three models that were pre-trained with the MS MARCO passage collection:

- doc2query [8]: This model is a sequence-to-sequence transformer model [10]. Given a passage, the model is pre-trained to predict the top k queries using the top-k sampling technique proposed by Fan et al. [4]. A maximumn of 400 passage tokens and 100 query tokens were used to train the model. For our experiments, we expanded each passage with queries[2] generated by doc2query. Each passage was appended with 10 queries, as recommended by Nogueira et al. [8].
- docTTTTTquery [7]: This model takes advantage of T5 [9], a sequence-to-sequence tansformer model, and it is pre-trained to generate queries given a passage. The model was trained with a maximum input and output of 512 and 64 tokens respectively. In our work, we expanded each passage with queries[3] generated by docTTTTTquery model. We appended the top 40 sampling as recommended by Nogueira et al. [7] to each passage.
- DeepCT [3]: This is a BERT-based expansion method to identify passage terms that are likely to appear in relevant queries. Given a query and a passage, the model was pre-trained to estimate a weight for each term in the passage. The trained model can be used to estimate the term weights for any passage without the need of queries. For our experiments, we used the expanded passages[4] provided by DeepCT.

[2]https://github.com/nyu-dl/dl4ir-doc2query
[3]https://github.com/castorini/docTTTTTquery
[4]https://github.com/AdeDZY/DeepCT

For query expansion, we used RM3 approach implemented by Anserini [12, 13].

For reranking the retrieved candidate passages, we used two existing pre-trained reranker models, available at pygaggle[5], that were both trained with the MS MARCO passage collection. The first is monoBERT [5], a BERT-based relevance classifier model for query-passage pairs. The second is monoT5 [6], a T5 model fine-tuned to produce the words "false" or "true" based on whether the passage is relevant or not to the query. Both reranker models were trained with a maximum of 512 input tokens, i.e., the total tokens of both the query and passage given to the model are not more than 512 tokens.

We note that, in the reranking step, we opted to use *original* (unexpanded) passages and queries as they exhibited better performance in our preliminary experiments over the expanded ones.

2.2 Pre-TREC Experiments

Before TREC submission, we experimented with the combination of each passage expansion technique and each reranker model mentioned in Subsection 2.1. For retrieval, we used BM25, implemented by Anserini, with the parameters set as recommended by Yilmaz et al. [14] to k_1=0.82 and b=0.68. An exception is the retrieval from the index of DeepCT-based expanded passages; we set the BM25 parameters to k_1=18 and b=0.7, as recommended by Dai and Callan [3].

For evaluation purposes, we used TREC-DL 2019 queries in our pre-TREC experiments [2]. In Table 1, we present the results of the full ranking subtask. We observe that when expanding the passages with the sequence-to-sequence based models, namely doc2query or docTTTTTquery, BERT reranker performed slightly better than T5 reranker in terms of NDCG@10. However, T5 outperformed BERT reranker, when passages were expanded with DeepCT [3], a BERT-based model.

Passage Expansion	Reranker	R@1000	NDCG@10
doc2query	BERT	0.7803	0.7266
	T5	0.7803	0.7257
docTTTTTquery	BERT	**0.8542**	**0.7476**
	T5	**0.8542**	**0.7391**
DeepCT	BERT	0.7946	0.7392
	T5	**0.7946**	**0.7404**

Table 1: Performance of document expansion and reranking models in Pre-TREC experiments for full passage ranking.

2.3 Submitted Runs

Based on our Pre-TREC experimental results, we selected the top 3 performing methods, highlighted in bold in Table 1, in terms of both NDCG@10 and Recall@1000, as our official runs submitted to the full ranking subtask:

- **bigIR-T5-BERT-F:** Using docTTTTTquery model for passage expansion, RM3 for query expansion, BM25 for retrieval, and monoBERT model for reranking.

[5]https://github.com/castorini/pygaggle

- **bigIR-T5xp-T5-F:** Using docTTTTTquery model for passage expansion, RM3 for query expansion, BM25 for retrieval, and monoT5 model for reranking.
- **bigIR-DCT-T5-F:** Using DeepCT model for passage expansion, RM3 for query expansion, BM25 for retrieval, and monoT5 model for reranking.

For the reranking subtask, we submitted two runs, one of each reranker model discussed in Section 2.1.

- **bigIR-BERT-R:** Using monoBERT for reranking.
- **bigIR-T5-R:** Using monoT5 for reranking.

2.4 Official TREC Results

The official results of our submitted runs for the passage retrieval task are presented in Table 2. We compare the performance of our runs against TREC-DL 2020 median which represents the mean of median per-topic scores.

As shown in Table 2, all our runs for both subtasks scored above the median. For the full ranking subtask, we observe that using DeepCT, the BERT-based approach for passage expansion, achieved better performance than using docTTTTTquery, the T5-based approach, achieving NDCG@10 of 0.7173 and 0.7034 respectively. The results also show that for full ranking runs that exploited the passages expanded with docTTTTTquery, the T5 reranker could not beat the BERT reranker.

Subtask	Run	NDCG@10
	TREC2020-Median	0.681
Full Ranking	bigIR-T5-BERT-F	0.7073
	bigIR-DCT-T5-F	**0.7173**
	bigIR-T5xp-T5-F	0.7034
Reranking	bigIR-BERT-R	**0.7201**
	bigIR-T5-R	0.7138

Table 2: NDCG@10 scores of the submitted passage retrieval runs compared to TREC-DL 2020 median score.

3 DOCUMENT RETRIEVAL TASK

In this Section, we present our approach and experimental results for the document retrieval task. We present our general approach in Section 3.1. We present our official submitted runs for both the full ranking and reranking subtasks in Section 3.2. Finally, we discuss the results of our official runs in Section 3.3.

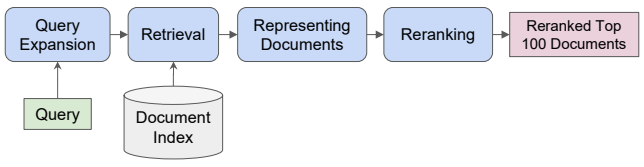

Figure 2: Proposed approach for document retrieval task.

3.1 Approach

Similar to the passage retrieval task, we used Anserini for indexing the document collection, and its BM25 implementation and RM3 query expansion for retrieving the initial set of candidate relevant documents for the full ranking subtask. For both full ranking and reranking subtasks, we adopted the monoT5 reranker model to rerank the candidate documents for better retrieval quality. Since the input sequence to monoT5 is limited in size (maximum 512 tokens), we experimented with three different representations of the document as input sequence to the model: title of the document, the head segment of the document (the leading 384 terms specifically, as recommended by Yan et al. [11]), and the concatenation of both. Figure 2 illustrates our approach.

3.2 Submitted Runs

We submitted three runs for the full ranking subtask. In all of our submitted runs, we set Anserini to use the BM25 retrieval model with RM3 query expansion, where k_1 and b were set to 0.82 and 0.68 respectively. For reranking the initial candidate set, we used different document representations for each submitted run as follows:

- **bigIR-DT-T5-F**: Using the title to represent each candidate document.
- **bigIR-DH-T5-F**: Using the head segment to represent each candidate document.
- **bigIR-DTH-T5-F**: Using the concatenation of both title and head segment to represent each candidate document.

We also submitted three runs to the reranking subtask as follows.

- **bigIR-DT-T5-R**: Using the title to represent each given document.
- **bigIR-DH-T5-R**: Using the head segment to represent each given document.
- **bigIR-DTH-T5-R**: Using the concatenation of both title and head segment to represent each candidate document.

3.3 Official TREC Results

Table 3 shows the performance results that our runs achieved. We notice that using only the title of the candidate documents, along with the query, as input to the reranker model did not perform well. In fact, it falls well below the TREC-DL 2020 median in both subtasks. This suggests an inadequate context for reranking, and can be explained by the fact that titles of documents do not usually contain sufficient information on the document content. The results also show that using the head segment of the document only performed slightly better than the median. However, using both the title and the head segment yielded a significant improvement in performance over the median.

While we opted to experiment with very simple representation of documents in our submitted runs, due to time limitation, there are several other ways of representing documents that we plan to study in the future.

4 CONCLUSION

In this paper, we present our bigIR group's first participation in the passage and document retrieval tasks at the TREC deep learning

Subtask	Run	NDCG@10
	TREC2020-Median	0.5733
Full Ranking	bigIR-DT-T5-F	0.539
	bigIR-DH-T5-F	0.5734
	bigIR-DTH-F	**0.5907**
Reranking	bigIR-DT-T5-R	0.5455
	bigIR-DH-T5-R	0.5846
	bigIR-DTH-T5-R	**0.6031**

Table 3: NDCG@10 scores of the submitted document retrieval runs compared to TREC-DL 2020 median score.

track 2020. We focused on simple ideas this year, such as document and query expansion. We explored different pre-trained models for passage/document expansion and reranking, including doc2Query, docTTTTTquery, DeepCT, monoBERT and monoT5. We conducted a preliminary study on TREC-DL 2019 data, based on which we selected the configuration of our submitted runs. All of our submitted runs for passage ranking and most of the submitted runs for document ranking outperformed the TREC median for the different subtasks. The results demonstrate two messages. First, monoBERT performed slightly better than monoT5 when passages were expanded with sequence-to-sequence based models; however, it could not beat monoT5 when passages were expanded using a BERT-based model. Second, using both the title and the head segment of the document at the reranking step for the document retrieval task significantly improved the results compared to each separately.

ACKNOWLEDGMENTS

This work was made possible by NPRP grant# NPRP 11S-1204-170060 from the Qatar National Research Fund (a member of Qatar Foundation). The work of Fatima Haouari was supported by GSRA grant# GSRA6-1-0611-19074 from the Qatar National Research Fund. The statements made herein are solely the responsibility of the authors.

REFERENCES

[1] Nasreen Abdul-Jaleel, James Allan, W Bruce Croft, Fernando Diaz, Leah Larkey, Xiaoyan Li, Mark D Smucker, and Courtney Wade. 2004. UMass at TREC 2004: Novelty and HARD. *Computer Science Department Faculty Publication Series* (2004), 189.

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

[3] Zhuyun Dai and Jamie Callan. 2019. Context-aware sentence/passage term importance estimation for first stage retrieval. *arXiv preprint arXiv:1910.10687* (2019).

[4] Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833* (2018).

[5] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. *arXiv preprint arXiv:1901.04085* (2019).

[6] Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. 2020. Document ranking with a pretrained sequence-to-sequence model. *arXiv preprint arXiv:2003.06713* (2020).

[7] Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019. From doc2query to docTTTTTquery. *Online preprint* (2019).

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

[9] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research* 21, 140 (2020), 1–67.

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.

[11] Ming Yan, Chenliang Li, Chen Wu, Bin Bi, Wei Wang, Jiangnan Xia, and Luo Si. 2019. IDST at TREC 2019 Deep Learning Track: Deep Cascade Ranking with Generation-based Document Expansion and Pre-trained Language Modeling.. In *TREC*.

[12] Peilin Yang, Hui Fang, and Jimmy Lin. 2017. Anserini: Enabling the use of Lucene for information retrieval research. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1253–1256.

[13] Peilin Yang, Hui Fang, and Jimmy Lin. 2018. Anserini: Reproducible ranking baselines using Lucene. *Journal of Data and Information Quality (JDIQ)* 10, 4 (2018), 1–20.

[14] Zeynep Akkalyoncu Yilmaz, Shengjin Wang, and Jimmy Lin. 2019. H2oloo at trec 2019: Combining sentence and document evidence in the deep learning track. In *Proceedings of the Twenty-Eighth Text REtrieval Conference (TREC 2019)*.