PASH at TREC 2020 Deep Learning Track: Dense Matching for Nested Ranking

Yixuan Qiao[1,*,†], Hao Chen[1,*], Liyu Cao[1,*], Liping Chen[1,2,*], Pengyong Li[1,3], Jun Wang[1], Peng Gao[1], Yuan Ni[1], Guotong Xie[1,4,5,†]

Abstract

This paper describes our participation in the passage ranking task of TREC 2020 Deep Learning Track. We propose a Dense retriever by a BERT-based dual-encoder framework utilizing in-batch negatives corresponding to a list-wise ranking loss. To add an extra degree of difficulty, we redesign the pre-training tasks of BERT to absorb additional information rendered by Dense Retriever. After pre-trained with general knowledge and document-level data, we firstly fine-tune it with a strictly balanced binary data using a point-wise ranking strategy. Then we re-rank the top k passages using a fine-grained data by gradual unfreezing skill which form a Nested Ranking framework. In addition, further combined with traditional retrieval methods and ensemble learning, we obtain the competitive ranking results.

Keywords: dense retrieval; transfer learning; pre-trained language model; multi-stage ranking

1 Introduction

In real-world scenarios, queries issued by users and passages compiled by editors are likely to use different styles of expressions for presenting the same meaning . It is the difference of thinking patterns that contribute to the well known query-passage mismatch problem. Traditional term-based methods, for instance different variants of TF-IDF and BM25 which can be viewed as representing the queries and passages as high-dimensional, sparse vectors (with weighting) are fundamentally powerless in cases where terms from the query and the passages don't match at all. Many studies have explored enriching query representations[1] or passage representations[2], latent representation learning[3], matching function learning[4], and so forth which inspire us to learn a dense representation of query and passage using a dual-encoder framework for matching. One natural

* Equal contribution.
1 Ping An Health Technology, Beijing, China.
2 Northeastern University, Shenyang, China.
3 Tsinghua University, Beijing, China.
4 Ping An Health Cloud Company Limited., Shenzhen, China.
5 Ping An International Smart City Technology Co., Ltd., Shenzhen, China.
†Corresponding Author. Email: {qiaoyixuan528,xieguotong}@pingan.com.cn.

idea is to endow transformer-based structure especially BERT[5] as encoders due to their effectiveness across a range of domains like Natural Language Processing, Computer Vision[6] and Reinforcement Learning.

As such, it is quite natural that a wealth of research has been dedicated to making remarkable improvements to the BERT model over the past few years. Many of them redesign the Next Sequence Prediction (NSP) task deemed ineffective since its lack of difficulty compared to Masked Language Model (MLM) task. Some of them significantly improving performance on downstream tasks that require reasoning about the relationship between sentence pairs[7, 8]. But the fact easily ignored is, the available information to MLM has also changed accordingly which rarely analysised. To capture knowledge tailored for ranking task, we propose the Retrieval-Augmented Pre-training strategy, which augments NSP with a Dense Retriever. Our goals are clear, they include: (a) relieve the pressure of storing a surprising amount of world knowledge implicitly in the parameters of a ever-larger network; (b) allow the model to retrieve and attend over information really need; (c) explicitly contain the retrieved segment as a new category greatly bridge the gap between pre-training and fine-tuning.

Our approach turn ranking into a relevance classification problem where we sort candidate passages p_i by $P(relevant = 1|q, p_i)$ for a given query q in a retrieve-and-rank paradigm. An obvious extension of this design is to employ a multi-stage cascade architecture. A common practice often consists of multi-way matching[9] and multi-stage ranking[10] which exists widely in many industrial systems such as Search Engines and Recommender Systems. In addition to term-based BM25 and semantic-based Dense Retriever, we also implemente string-based method operates on word composition after stemming that measures similarity between query and passage for approximate string matching or comparison. In the course of second stage ranking, we transfer the knowledge of retrieval augmented ranker to a fine-grained re-ranker using annotated query passage pairs from Test set 2019. The re-ranker is applied in top K passages of ranked list generated by the ranker to get the final ranking result. For passage re-ranking task, we submit runs by integrating thirteen basic re-rankers and seven other attempts. For passage full ranking task, we further adopt the multi-way matching technique.

The rest of the paper is organized as follows: section 2 will introduce the details of our approach, including Dense Retriever, Retrieval Augmented Ranker and Fine-grained Re-ranker, section 3 and 4 will give experiment results and conclusion.

2 Methodology

In this section, we sequentially describe the component in the nested ranking pipeline. The passage ranking task contains 1,010,916 queries on a collection of 8,841,823 passages, totally 532,761 query passage pairs annotated as positive for relevance. Few queries are matched with multiple relevant passages. NDCG metric with 3-level or 4-level judgements is the main official evaluation index.

2.1 Dense Retriever

Traditional information retrieval methods such as BM25 tend to focus on term-based matching which difficult in retrieving semantically correct answers. Inspired by DPR[11], we learn the dense representation of query and passage by a dual-encoder framework capturing the semantic similarity in latent space which makes up the deficiency of the traditional sparse representation.

Encoders We use two independent BERT[5] models from Google pre-trained checkpoint(base, uncased) as initialization and use the output of [CLS] token to represent query or passage.

Training data & strategy We apply an sample-efficient list-wise loss function in the case of only possess positive pairs.

$$\begin{aligned} L\left(q_i, p_1, p_2, \cdots, p_i, \cdots, p_n\right) &= -\log \frac{e^{\operatorname{sim}(q_i, p_i)}}{\sum_{j=1}^{n} e^{\operatorname{sim}\left(q_i, p_j\right)}} \\ \operatorname{sim}(q_i, p_j) &= \mathrm{BERT}_Q(q_i)^\top \, \mathrm{BERT}_P(p_j) \end{aligned} \tag{1}$$

This loss expects as input a batch consisting of query and passage pairs $(q_i, p_1), (q_i, p_2), \ldots, (q_i, p_n)$ where we assume that (q_i, p_i) are a positive pair and (q_i, p_j) for $i \neq j$ a negative pair. Moreover, for each given query, we add a negative passage retrieved by BM25 with highest score to improve performance. We choose simple inner product similarity function for fast indexing provided by FAISS[12].

2.2 Retrieval Augmented Ranker

In this subsection, we describe the data and basic setup during the pre-training stage and fine-tuning stage of Ranker in detail.

Pre-training Stage We modify the pre-training tasks of BERT to absorb additional information rendered by Dense Retriever. Many previous studies argue that Next Sentence Predict (NSP) task is rather simplistic even useless so we switch it to a relatively difficult three-class classification task. Specifically, given a pair of segments (S_1, S_2) as input, we predict whether S_2 is the retrieved segment closest to S_1 using Dense Retriever, or the next sentence that follow S_1, or a random segment from a different document, under equal chance. Combined with original Masked Language Model (MLM) task, we firstly use BooksCorpus and English Wikipedia (19G) and then Document ranking dataset (22G) provided by TREC official as pre-training data to sequentially train the 24-layer BERT large architecture ($L = 24, H = 1024, A = 16$) optimized through mixed precision and distributed training. The latter uses the checkpoint of the former as initialization. We further use the LAMB optimizer that helps accelerate training using large minibatches. For each pre-training corpus, the model is pre-trained for approximately 8k updates with minibatches containing 65536 examples of maximum length 384 tokens. The whole pre-training process is performed on a distributed computing cluster consisting of 32 Telsa V100 GPU cards which takes about 15 days.

Fine-tuning Stage We use a trivial method constructing the training data for fine-tuning stage. In particular for each query that already has a positive (relevant) passage, we randomly sample a negative (not relevant) passage from passage pool composing a strictly balanced data set containing about 500k query-passage pairs. We represent the input sentence pairs as described in BERT, and use the final hidden vector $C \in \mathbb{R}^H$ corresponding to the first input token ([CLS]) as the aggregate representation. The only new parameters introduced during fine-tuning are classification layer weights $W \in \mathbb{R}^{K \times H}$, where K is set to 2. We compute a point-wise loss with C and W, i.e., $\log\left(\operatorname{softmax}\left(CW^T\right)\right)$, fine-tune for 5 epochs with a learning rate of 2e-5 and a batch size of 32 performed on 4 Telsa V100 GPU cards which takes about 30 hours.

After fine-tuning the model, we run it on unseen queries and compute a relevance score for each candidate Top 1000 passage, and afterwards simply sort them in order to obtain the first phase of ranking results.

2.3 Passage Re-ranking & Ensemble

During Fine-tuning, we just have ground-truth label {0,1} namely {relevant, not relevant} and there is no discrimination among relevant passages. However, NIST refines it into three levels, namely perfect related, highly relevant and related leading to our Ranker suffers from a finetune-inference discrepancy. Therefore, we propose a fine-grained re-ranker acted on top K passages after Ranker to alleviate the mismatch status.

Fine-grained Re-ranker We employ the annotated query passage pairs from Test 2019 as training data. With the fine-tuned checkpoint of Ranker as a staring point, we replace the last layer with a random initialized 4-class classification layer while keeping other layers unchanged. Inspired by Gradual unfreezing[13], we first unfreeze the last layer and fine-tune all unfrozen layers for one epoch. We then unfreeze the next lower frozen layer and repeat, until we fine-tune all layers until convergence at the last iteration. After full training, we apply it to top K passages ranked by Ranker to get the refined ranking result where K is selected based on the development set ranking accuracy. The score of the Re-ranker is added to the score of the previous Ranker.

Ensemble Diversity, that is, the difference among the individual learners is a fundamental issue in ensemble methods. Our ensemble strategy is based on average of probability scores from twenty Re-rankers with equally-weighted which acquires the best accuracy on the TEST 2019. Thirteen of them are initialized with different random seeds, others take different model structures or training data.

- On the aspect of the model structure, we also fine-tune ALBER-xxlarge, ELECTRA-Large and XLNet-Large model using the same training data as Ranker.
- We also use some novel but not powerful training data selected from a variety of attempts in the course of experiment, they include:
    a) Change the order of query and passage in the input representation.
    b) Inspired by [2], use T5[14] as the expansion model to generate 5 pseudo queries then append to the original passage.
    c) Expand query by appending 3 queries generated by GNMT[15] which was trained to rephrase an input query into semantically similar queries.
    d) Delete the words in passage that appear in query except stop words.

Although the performance of the above seven models are not as well performance as the Re-ranker, they do play an important role in final ranking.

2.4 Passage Full Ranking & Ensemble

To effectively address the query-passage mismatch challenge, we absorb the quintessence of the complementary characteristics in traditional and deep semantic matching methods before ranking and re-ranking.

Multi-way Matching In our experiments, we use BM25+PRF[16] with tuned parameters $k_1 = 0.82$, $b = 0.68$. Compared to unigram features (i.e., on individual terms) such as BM25 scores, many n-gram features are better signals of relevance, but also more computationally expensive to compute, in both time and space. Therefore, we use string-based method compare the n-grams g_n with n being 2, 3, and 4 between query and passage. Specifically, for (p_i, q_i), we divide q_i into several n-grams (consecutive word) with length n, and to determine whether g_n is included in p_i, a match scoring function can be written as:

$$
\begin{aligned}
sim(p_i, q_i) &= \exp(\sum_{n=2}^{4} w_n \log p_n) \\
p_n &= \frac{\sum_{g_n \in q_i \bigcap g_n \in p_i} count_{dist}(g_n)}{\sum_{g_n \in q_i} count(g_n)}
\end{aligned} \tag{2}
$$

Where $count_{dist}$ means that the same n-gram is calculated only once, and w_n is the weight of n-gram set to $1/3$. A linear combination of multi-way matching scores or completely ignore scores further boosts retrieval effectiveness.

Ensemble Benefited from multi-way matching, we just sequentially train Rankers and Re-rankers with ten different random seeds for ensemble learning.

3 Results

We submitted six official runs in both passage re-ranking and full ranking subtasks. Table 1 and 2 present the results of our all runs in which the former judges the passage on a four-point scale of Not Relevant (0), Related (1), Highly Relevant (2), and Perfect (3). Since "related" means that the passage is on-topic, but don't actually answer the question, it is treated as NOT relevant by Table 2. We use pash_r∗ to denote the runs for passage re-ranking, pash_f∗ for passage full ranking.

Table 1: Ranking performance with 4-level judgements.

RUN	RUN Description	RNDCG	NDCG@10	NDCG@1000
pash_r1	Single Model	0.6556	0.7463	0.6797
pash_r2	Ensemble Model	0.6919	0.8011	0.7020
pash_r3	Ensemble Model	0.6956	0.8031	0.7034
pash_f1	Single Model	0.6912	0.7956	0.7074
pash_f2*	Single Model	0.6866	0.7941	0.7019
pash_f3	Single Model	0.6977	0.8005	0.7120

* Due to our carelessness, pash_f2 is the same as pash_f1 just using a different random seed.

Table 2: Ranking performance with 3-level judgements.

RUN	RUN Description	AP	R@1000	NDCG@10	NDCG@1000
pash_r1	Single Model	0.4969	0.7526	0.6896	0.6835
pash_r2	Ensemble Model	0.5420	0.7526	0.7589	0.7153
pash_r3	Ensemble Model	0.5445	0.7526	0.7628	0.7189
pash_f1	Single Model	0.5455	0.7667	0.7504	0.7186
pash_f2*	Single Model	0.5389	0.7595	0.7513	0.7146
pash_f3	Single Model	0.5504	0.7708	0.7561	0.7252

* Due to our carelessness, pash_f2 is the same as pash_f1 just using a different random seed.

From the results, we can clearly see the remarkable performance of pash_r1 owing to retrieval augmented pre-training and nested ranking strategy. Under same circumstances, equipped with

Dense Retriever (pash_f1) is even comparable to the best ensemble re-ranking results (pash_r3), which demonstrates the effectiveness of our dual-encoder framework by leveraging in-batch negatives. With the help of more different model strucures and training data forms, the experiments show that ensemble method can effectively improve performance of the ranking.

Table 3 and 4 shows the detailed performace distribution of pash_r3 and pash_f3 according to AP, NDCG@10, NDCG@1000 and RR. Other runs are provided in Appendix. The per-topic minimum, maximum, and median scores are achieved across 59 submitted runs from contestants. From the results, we observe that a large proportion of ranking predictions fall into the median to best region. Specifically, 87.0%, 90.7%, 79.6%, 88.9% for re-ranking subtask and 90.7%, 88.9%, 85.2%, 88.9% for full ranking subtask in terms of the four metrics respectively.

Table 3: Performace distribution of pash_r3.

Evaluation	AP	NDCG@10	NDCG@1000	RR
Best	7	11	4	48
Best to Median	40	38	39	0
Median	4	3	4	5
Median to Worst	3	2	7	1
Worst	0	0	0	0

Table 4: Performace distribution of pash_f3.

Evaluation	AP	NDCG@10	NDCG@1000	RR
Best	4	14	5	46
Best to Median	45	34	41	2
Median	2	4	2	4
Median to Worst	3	2	6	2
Worst	0	0	0	0

4 Conclusion

In this paper, we propose the dense retriever and retrieval augmented ranker for ad-hoc passage matching. We demonstrate that semantically-rich dense representation can outperform traditional term-based and string-based retrieval method. Multi-way matching is put into effect benefited from complementarity among them for best accuracy. We modify the pre-training task by giving model the ability to learn from additional retrieval augmented information especially to tackle ranking task. Fine-tuned with coarse and fine grained data make up our nested ranking framework. The experimental results show the effectiveness and high performance of our method. We believe that design highly transferable pre-training strategy will be a focus point of future work.

Appendix

Table 5: Performace distribution of pash_r1.

Evaluation	AP	NDCG@10	NDCG@1000	RR
Best	1	6	0	43
Best to Median	41	37	42	4
Median	5	6	3	5
Median to Worst	7	5	9	2
Worst	0	0	0	0

Table 6: Performace distribution of pash_r2.

Evaluation	AP	NDCG@10	NDCG@1000	RR
Best	6	12	4	47
Best to Median	41	36	39	0
Median	4	3	5	5
Median to Worst	3	3	6	2
Worst	0	0	0	0

Table 7: Performace distribution of pash_f1.

Evaluation	AP	NDCG@10	NDCG@1000	RR
Best	4	12	4	44
Best to Median	44	36	41	1
Median	3	3	3	5
Median to Worst	3	3	6	4
Worst	0	0	0	0

Table 8: Performace distribution of pash_f2.

Evaluation	AP	NDCG@10	NDCG@1000	RR
Best	4	11	4	46
Best to Median	45	37	41	1
Median	3	4	3	5
Median to Worst	2	2	6	2
Worst	0	0	0	0

References

[1] Zerveas, G., Zhang, R., Kim, L. & Eickhoff, C. Brown university at trec deep learning 2019. In *Proceedings of the Twenty-Eighth Text REtrieval Conference, TREC* (2019).

[2] Nogueira, R., Yang, W., Lin, J. & Cho, K. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375* (2019).

[3] Huang, P.-S. *et al.* Learning deep structured semantic models for web search using clickthrough data.

In *Proceedings of the 22nd ACM International Conference on Information amp; Knowledge Management*, CIKM '13, 2333–2338 (Association for Computing Machinery, New York, NY, USA, 2013).

[4] Nogueira, R. & Cho, K. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085* (2019).

[5] Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171–4186 (2019).

[6] Carion, N. *et al.* End-to-end object detection with transformers. *arXiv preprint arXiv:2005.12872* (2020).

[7] Wang, W. *et al.* Structbert: Incorporating language structures into pre-training for deep language understanding. In *International Conference on Learning Representations* (2019).

[8] Lan, Z. *et al.* Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations* (2019).

[9] Wang, Z. *et al.* Cold: Towards the next generation of pre-ranking system. *arXiv preprint arXiv:2007.16122* (2020).

[10] Nogueira, R., Yang, W., Cho, K. & Lin, J. Multi-stage document ranking with bert. *arXiv preprint arXiv:1910.14424* (2019).

[11] Karpukhin, V. *et al.* Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906* (2020).

[12] Johnson, J., Douze, M. & Jégou, H. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data* (2019).

[13] Howard, J. & Ruder, S. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 328–339 (2018).

[14] Raffel, C. *et al.* Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research* **21**, 1–67 (2020).

[15] Wu, Y. *et al.* Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144* (2016).

[16] Zeng, Z. & Sakai, T. Bm25 pseudo relevance feedback using anserini at waseda university. In *OSIRRC@ SIGIR*, 62–63 (2019).