TUA1 at the TREC 2019: Deep Learning Track

Yun Gao[1] , **Hai-Tao Yu**[2] , **Xin Kang**[1] and **Fuji Ren**[1]
[1] Tokushima University, Japan
[2] University of Tsukuba, Japan
c501837010@tokushima-u.ac.jp, yuhaitao@slis.tsukuba.ac.jp, {kang-xin, ren}@is.tokushima-u.ac.jp

Abstract

This paper details our participation in the TREC 2019 Deep Learning Track task including Passage Ranking task. In the Top-1000 Re-ranking subtask of Passage Ranking task, we performed passage ranking based on the technique of Conv-KNRM and BERT. In order to get a better ranking result, we divided the task into two stages. In the first stage we use Conv-KNRM model for initial ranking, then in the second stage we combine the results with a state-of-the-art BERT re-ranker.

Team Name

TUA1

Subtask

Top-1000 Re-ranking

Keywords

Conv-KNRM, BERT re-ranking, Learning to rank

1 Introduction

The TUA1 group from The University of Tokushima participated in Passage Ranking task of the TREC 2019 Deep Learning Track Task, i.e., the Passage Ranking Top-1000 Re-ranking subtask. For detailed introductory information of this subtask, please refer to the task homepage [1], the TREC Tracks homepage[2], and the overview of the TREC 2019 Deep Learning Paper [Craswell *et al.*, 2019]. The Deep Learning Track has two tasks: Passage Ranking and Document Ranking. The passage ranking task has a full ranking and re-ranking subtasks. We participate in top-1000 re-ranking subtask. This subtask provides with an initial ranking of 1000 passages and re-ranks these passages based on their likelihood of containing an answer to the question. We investigated the traditional IR model, such as BM25 [Robertson *et al.*, 1994], neural IR model and BERT [Devlin *et al.*, 2018]. In this task, we use the most advanced neural retrieval model Conv-KNRM [Dai *et al.*, 2018] and BERT to rank passages. This article will elaborate on our experimental process and results.

In the remainder of this paper, we outline the Deep Learning Track tasks in Section 2. Section 3 and Section 4 details the approaches proposed for top-1000 re-ranking subtask respectively.We conclude our work in Section 5.

[1]https://microsoft.github.io/TREC-2019-Deep-Learning/

[2]https://trec.nist.gov/tracks.html

2 Deep Learning Track Tasks

The deep learning track has two tasks: Passage Ranking and Document Ranking. Both tasks use a large human-generated set of training labels, from the MS-MARCO[3] dataset.

2.1 Passage Ranking Task

The passage ranking task has a full ranking and re-ranking subtasks.In context of full ranking subtask, given a question to rank passages from the full collection in terms of their likelihood of containing an answer to the question, and submit up to 1000 passages for this end-to-end retrieval task.

2.2 Top-1000 re-ranking subtask

In context of top-1000 re-ranking subtask, an initial ranking of 1000 passages is provided and expected to re-rank these passages based on their likelihood of containing an answer to the question. In this subtask, we can compare different re-ranking methods based on the same initial set of 1000 candidates, with the same rationale as described for the document re-ranking subtask.

3 Method

We used the training set provided by the organizer to train a Conv-KNRM model, and then used it to rank in the first stage. After that, we used a BERT pre-training model, and then re-ranked in the second stage to get the final ranking result. An overview of the proposed method is shown in Figure 1.

3.1 Conv-KNRM

Conv-KNRM is a Convolutional Kernel-based Neural Ranking Model that models n-gram soft matches for ad-hoc search. It uses Convolutional Neural Networks to represent n-grams

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

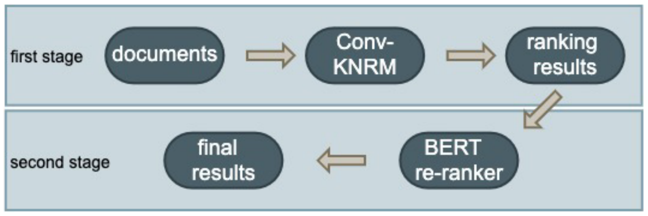

Figure 1: Given documents, Conv-KNRM model predicts document ranking results, acorrding the ranking results, BERT re-ranker predicts the final ranking results.

of various lengths and soft matches them in a unified embedding space. The n-gram soft matches are then utilized by the kernel pooling and learning-to-rank layers to generate the final ranking score. Conv-KNRM can be learned end-to-end and fully optimized from user feedback.The architecture of our Conv-KNRM model is identical to the base model described in [Dai *et al.*, 2018], we used train.triple.small.tsv of the MS-MARCO dataset as the training set. It has three columns of data, namely query, positive passage and negative passage. We train with a batch size of 64 tokens, and vocab size of 851354.

3.2 BERT Re-rank

The re-ranker is to estimate a score s_i of how relevant a candidate passage d_i is to a query q. We use BERT as our re-ranker. Using the same method used by [Nogueira *et al.*, 2019].The query as sentence A and the passage text as sentence B. We use a $BERT_{LARGE}$ model as a binary classification model. Using the [CLS] vector as input to a single layer neural network to obtain the probability of the passage being relevant. We independently calculate the probabilities of each article and rank them according to these probabilities to obtain the final list of passages. We optionally re-rank these retrieved documents using BERT described in [Nogueira and Cho, 2019].

3.3 Conv-KNRM + BERT re-rank

First, a large number of possibly relevant documents to a given question are retrieved from Conv-KNRM Model. In the second stage, passage re-ranking, each of these documents is scored and re-ranked by a more computationally-intensive method.

We retrieve documents as in Conv-KNRM and further re-rank the documents with BERT.

4 Experimental Setup

4.1 Data

To train and evaluate the models, we use the following dataset:
MS MARCO is a passage re-ranking dataset with 8.8M passages obtained from the top-10 results retrieved by the Bing search engine [Nguyen *et al.*, 2016]. The training set contains approximately 500k pairs of query and relevant documents. On average each query has one relevant passage. The development and test sets contain approximately 6,900 queries each, but relevance labels are made public only for the development set.

The training set contains approximately 400M tuples of a query, relevant and non-relevant passages. The development set contains approximately 6,900 queries, each paired with the top 1,000 passages retrieved with BM25 from the MS MARCO corpus.

An evaluation set with approximately 6,800 queries and their top 1,000 retrieved passages without relevance annotations is also provided.

4.2 Ranking Methods

We evaluate the following ranking methods:
BM25: We use the Anserini open-source IR toolkit [4] (Yang et al., 2017, 2018) to index the original (non-expanded) documents and BM25 to rank the passages. During evaluation, we use the top1000 re-ranked passages.
Conv-KNRM: We use the open-source Neural Kernel Match IR method Kernel-Based-Neural-Ranking-Models [5] to train a Conv-KNRM model.
BERT re-ranker: We use the open-source dl4marco-bert [6] as our re-ranker.
Conv-KNRM+BERT: The ranking result of Conv-KNRM is put into BERT re-ranker to obtain the final ranking result.

4.3 Metrics

To evaluate the effectiveness of the methods on MS MARCO, we use its official metric, mean reciprocal rank of the top-10 documents (MRR@10). For TREC Deep Learning task, we use normalized discounted cumulative gain (NDCG), precision (P), and mean average precision (MAP).

4.4 runs

We submitted one run for the re-ranking subtask.

- TUA1-1. We combine Conv-KNRM results with BERT re-ranker, achieve our team's best results.

4.5 Experimental Results

Table 1 shows the performance on the MS-MARCO dev set, where MMR@l is used with a cut-off value of 10.

	MRR@10
BM25	0.184
Conv-KNRM	0.224
Conv-KNRM + BERT	0.366

Table 1: Mean reciprocal rank scores for MS-MARCO dev set.

Table 2 shows the performance of the run, where P@l and NDCG@l are used as the evaluation metrics with a cut-off value of 10. P@l represents the precision at l, NDCG@l represents the normalised discounted cumulative gain.MAP represents the mean average precision.

[4]https://github.com/castorini/anserini
[5]https://github.com/thunlp/Kernel-Based-Neural-Ranking-Models
[6]https://github.com/nyu-dl/dl4marco-bert

Run	NDCG@10	P@10	MAP
TUA1-1	0.7314	0.6372	0.4571

Table 2: TREC evaluation scores for re-rank subtask.

5 Conclusions

In this paper, we described our approaches to solve the Top-1000 Re-ranking subtask in the TREC 2019 Deep Learning Track task. Our current implementation is based on integrating three open-source toolkits: Kernel-Based-Neural-Ranking-Models, Anserini(BM25), and dl4marco-bert(BERT re-ranker). Among the neural approaches, the best-performing methods tended to use transfer learning, employing a pre-trained language model such as BERT. Although the neural IR model has good results, there are also some limitations of our current work. For example, (1) the low-frequency term is a recurring challenge for information retrieval model, especially the neural IR framework is difficult to fully capture the word not often observed. (2) although BERT re-ranking have a good result, but it is very time-consuming. In the future work, we will investigate the effectiveness of word embedding on ranking results, and improve the speed of document re-ranking.

6 Acknowledgments

This research has been supported by JSPS KAKENHI Grant Number 19K20345 and Grant Number 19H04215.

References

[Craswell *et al.*, 2019] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. Overview of the trec 2019 deep learning track. In *Proceedings of the 28th TREC Conference*, 2019.

[Dai *et al.*, 2018] Zhuyun Dai, Chenyan Xiong, James P. Callan, and Zhiyuan Liu. Convolutional neural networks for soft-matching n-grams in ad-hoc search. In *WSDM*, 2018.

[Devlin *et al.*, 2018] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2018.

[Nguyen *et al.*, 2016] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. Ms marco: A human generated machine reading comprehension dataset. *ArXiv*, abs/1611.09268, 2016.

[Nogueira and Cho, 2019] Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with bert. *ArXiv*, abs/1901.04085, 2019.

[Nogueira *et al.*, 2019] Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. Document expansion by query prediction. *ArXiv*, abs/1904.08375, 2019.

[Robertson *et al.*, 1994] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. Okapi at trec-3. In *TREC*, 1994.