Longformer for MS MARCO Document Re-ranking Task

Ivan Sekulić[1], Amir Soleimani[2], Mohammad Aliannejadi[2], and Fabio Crestani[1]

[1] Faculty of Informatics
Università della Svizzera italiana
Lugano, Switzerland
name.surname@usi.ch
[2] University of Amsterdam
Amsterdam, The Netherlands
n.surname@uva.nl

Abstract. This technical report describes the approach of the Università della Svizzera italiana and the University of Amsterdam for MS MARCO document ranking task and TREC Deep Learning track 2020. Two step document ranking, where the initial retrieval is done by a classical information retrieval method, followed by neural re-ranking model, is the new standard. The best performance is achieved by using transformer-based models as re-rankers, e.g., BERT. We employ Longformer, a BERT-like model for long documents, on the MS MARCO document re-ranking task. The complete code used for training the model can be found on: https://github.com/isekulic/longformer-marco

Keywords: Document ranking · Longformer · Neural ranking.

1 Introduction

Document ranking is central problems in information retrieval (IR). Given a query, the task is to rank the documents of some collection so that the most relevant ones appear on top of the list. Recently, neural ranking models have shown superior performance compare to the traditional IR methods. Given the much higher computational need of neural models, the two step retrieval process is widely adapted. First, a traditional IR method, like BM25 or query likelihood, retrieves top k documents from a given collection. Then, a computationally expensive neural model re-ranks the top k documents from the initial retrieval step.

A number of neural models for ranking has been proposed in recent years. Some of them are: DRMM [4], KNRM [10], Co-PACRR [5], DUET [7], and Conformer-Kernel with QTI [8]. With a combination of new generation of neural models, namely the transformer architecture, and large-scale datasets, neural rankers arose as superior to traditional methods, which was not possible before [6]. Most notable model from the transformers family is probably BERT [2],

which has been very successfully applied to passage ranking [9], outperforming state-of-the-art by a large margin.

The largest dataset for the document ranking task is MS MARCO (Microsoft Machine Reading Comprehension). Transformer architecture, namely BERT, has already proven effective on the MS MARCO passage ranking task. However, the documents are much longer than the passages, making the task of document ranking more challenging.

To address the issue of increased length of the documents, we employ Longformer [1] – a BERT-like model for long documents. Longformer has an adjusted attention mechanism that combines local, BERT-like windowed attention, with a global attention, allowing the model to attend over much longer sequences than standard self-attention. Compared to BERT that typically processes up to 512 tokens at a time, Longformer is pre-trained on documents with length of 4096 tokens. We reach MRR@100 of 0.329 and 0.305 on the official dev and the test sets, respectively.

2 Dataset

We train and evaluate Longformer on MS MARCO document ranking dataset. It consists of more than 370k queries, 3.2 million documents, and the corresponding relevance labels for query-document pairs. Relevance labels are transferred from the MS MARCO passage ranking task, by mapping a positive passage to a document containing the passage, for each query. This is done under an assumption that the document that contains a relevant passage is a relevant document. Additional information about the dataset is present in Table 1.

Table 1. Number of documents in MS MARCO corpus and the number of queries in train, dev, and test set.

# documents		3.2M
# queries	train	367,013
	dev	5,193
	test	5,793

The document ranking task features two tasks:

Document Re-Ranking Given a candidate top 100 documents for each query, as retrieved by BM25, re-rank the documents by relevance.

Document Full Ranking Given a corpus of 3.2m documents generate a candidate top 100 documents for each query, sorted by relevance.

We participate in the document re-ranking task, where we use Longformer to assign relevance to the 100 documents retrieved by BM25, which are provided by the organizers.

3 Experiments

We train Longformer [1] to estimate relevance of a query-document pair. The training setting is formulated as in [9]. We feed the query as sentence A and the document as sentence B to the tokenizer, which yields the following input to the Longformer[3]:

$$<s> \; query \; </s> \; document \; </s>$$

We truncate the document such that the sequence of the concatenated query, document, and the separator tokens does not exceed 4096 tokens. After passing the sequence through the Longformer model, the <s> vector is given as input to a classifier head, consisting of two linear layers with dropout and a non-linear function. The classifier outputs a two-dimensional vector, where the first dimension indicates probability of a document not being relevant to the query, while the second indicates relevance probability. For a given query, we rank the candidate documents based on the relevance probability, which is computed independently for each document.

We fine-tune the pre-trained Longformer with a cross-entropy loss, using the following hyperparameters: batch size of 128, Adam optimiser with the initial learning rate of 3×10^{-5}, a linear scheduler with warmup for 2500 steps, trained for 150k iterations. Further hyperparameter tuning might yield better results. For training, we also reduce the positive to negative document ratio to 1:10, from the given 1:100 (as each query is given top 100 documents extracted by BM25 by the organisers). We use PytorchLightning [3] for our training setting implementation and HuggingFace's Transformers package [**?**] for Longformer implementation.

4 Results

4.1 TREC Deep Learning 2020

The results of our TREC Deep Learning submission are presented in Table 2. We participate in document re-ranking task with one run only.

Table 2. TREC Deep Learning 2020 results.

Run name	MRR	MAP	MAP@10
longformer_1	0.8889	0.3503	0.2028

[3] Tokens <s> and </s> are equivalent to the [CLS] and [SEP] tokens in BERT tokenizer, respectively.

4.2 MS MARCO document re-ranking

The results on the MS MARCO document re-ranking task on the dev and the test set are presented in Table 3. The official metric is mean reciprocal rank (MRR@100). Other submissions and approaches can be found on the official leaderboard[4].

Table 3. MRR@100 of the Longformer and the official baselines provided by the organisers. [8]

	dev	test
Indri Query Likelihood		0.192
Conformer-kernel with QTI (NDRM3)		0.293
Conformer-kernel with QTI (NDRM1)		0.307
Longformer	0.336	0.305

5 Conclusions

We employed Longformer for MS MARCO document re-ranking task. We submitted the model to the official MS MARCO leaderboard, as well as to the TREC Deep Learning track 2020. The results suggest that further work is required to match the performance of other transformer-based models.

References

1. Beltagy, I., Peters, M.E., Cohan, A.: Longformer: The long-document transformer. arXiv preprint arXiv:2004.05150 (2020)
2. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
3. Falcon, W.: Pytorch lightning. GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning **3** (2019)
4. Guo, J., Fan, Y., Ai, Q., Croft, W.B.: A deep relevance matching model for ad-hoc retrieval. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management. pp. 55–64 (2016)
5. Hui, K., Yates, A., Berberich, K., De Melo, G.: Co-pacrr: A context-aware neural ir model for ad-hoc retrieval. In: Proceedings of the eleventh ACM international conference on web search and data mining. pp. 279–287 (2018)
6. Lin, J.: The neural hype and comparisons against weak baselines. In: ACM SIGIR Forum. vol. 52, pp. 40–51. ACM New York, NY, USA (2019)

[4] https://microsoft.github.io/msmarco/#docranking

7. Mitra, B., Diaz, F., Craswell, N.: Learning to match using local and distributed representations of text for web search. In: Proceedings of the 26th International Conference on World Wide Web. pp. 1291–1299 (2017)
8. Mitra, B., Hofstatter, S., Zamani, H., Craswell, N.: Conformer-kernel with query term independence for document retrieval. arXiv preprint arXiv:2007.10434 (2020)
9. Nogueira, R., Cho, K.: Passage re-ranking with bert. arXiv preprint arXiv:1901.04085 (2019)
10. Xiong, C., Dai, Z., Callan, J., Liu, Z., Power, R.: End-to-end neural ad-hoc ranking with kernel pooling. In: Proceedings of the 40th International ACM SIGIR conference on research and development in information retrieval. pp. 55–64 (2017)