Improving Out-of-Distribution Generalization of Neural Rerankers with Contextualized Late Interaction

Anonymous ACL submission

Abstract

Recent advances in neural information retrieval based on pre-trained language models reveal that directly fine-tuning the `[CLS]` vector for downstream retrieval tasks might not yield a robust bi-encoder retriever on out-of-distribution (OOD) datasets. Therefore, many methods are proposed to increase OOD generalization, among which the multi-vector retrievers achieve the best balance between the in-domain and OOD effectiveness. In this paper, we explore whether late interaction, the building stone of multi-vector, is also helpful to neural *rerankers* that rely on the `[CLS]` vector alone to compute the similarity score. Although many would argue that the rerankers already gather the token-interaction information via the attention mechanism, we find adding late interaction still brings an extra 5% improvement "for free" on average on OOD datasets, with little increase in latency and no degradation in in-domain effectiveness. Extensive experiments show that this finding is consistent across different model sizes and first-stage retrievers, and that the improvement is more prominent on longer queries. Our findings suggest that for neural rerankers, boiling all information into the `[CLS]` token is not the optimal choice for all scenarios, and more studies are required to better utilize the reranker's structure.

1 Introduction

The two-stage retrieve-then-rerank pipeline has been the *de facto* design for many information retrieval systems. With the advancement in pre-trained language models, these retrieval systems also benefit from the rich semantics in the contextualized representations which could be fine-tuned for measuring the similarity between queries and documents. Commonly, the `[CLS]` token vector at the last layer is often chosen to be the sequence-level representation. However, neural retrievers that only use the `[CLS]` vector might be less robust on out-of-distribution (OOD) datasets as some of the token-level granularity might not be captured. Therefore, methods such as further pre-training or adding token-level interaction have been applied to improve the OOD generalization of the neural retrievers. Among them, late interaction models (Khattab and Zaharia, 2020; Gao et al., 2021a), also known as the multi-vector retrievers, strikes a perfect balance between the in-domain and OOD effectiveness among neural retrievers. This is usually credited to its design which takes the last layer of contextualized token embeddings to compute the final similarity other than just using the `[CLS]` vector. Given its powerful design, in this paper, we raise the following question:

Add LI?	**MS MARCO** MRR@10	**BEIR Avg.** nDCG@10	**Search Latency**
×	0.390	0.467	1.18s
✓	0.392	0.491	1.28s

Table 1: The in-domain score (on MS MARCO), OOD score (on BEIR), and search latency of rerankers w/o and w/ adding late interaction. Rerankers are initialized from MiniLM. **LI:** Late Interaction.

Can neural rerankers that only use the `[CLS]` vector for computing similarity scores also benefit from adding the late interaction?

Intuitively, many would argue that the attention mechanism at the previous layers already gathers the token-level interaction between the query and the document. However, in this paper, we show that late interaction at the last layer actually brings "free" OOD capacity to rerankers. As shown Table 1, after adding late the interaction, the averaged nDCG@10 on BEIR is improved by 5% (from 0.467 to 0.491), while the in-domain score (MRR@10 on MS MARCO) is not affected and the search latency is only slightly increased. We also show that this improvement is orthogonal to the better OOD capacity brought by larger model size, and consistent when reranking candidates from all categories of retrievers.

2 Related Work

Nogueira and Cho (2019) was one of the first work on cross-encoders, which serve to rerank a subset of documents returned from the "first-stage" retrievers. It considers retrieval as a classification task, and uses Transformer encoders following the formulation of the next sentence prediction (NSP) pretraining task in BERT, where only the `[CLS]` vector is used in classifying the (query, document) pair and computing the relevant score. Afterward, CEDR (MacAvaney et al., 2019) also proposes to integrate token information rather than use only `[CLS]`, but it processes token representations at all Transformer layers using the pre–BERT neural rerankers (Xiong et al., 2017; Guo et al., 2016; Hui et al., 2017), which is more complex in structure and adds higher computational overhead.

Recent works on first-stage retrieval have demonstrated the effectiveness of adding sparse information into dense retrieval. Chen et al. (2021) The combination of the token information and dense `[CLS]` vector could also be done explicitly, by either adding the scores computed from `[CLS]` and token information, or concatenated aggregated token vectors to the `[CLS]` vector (Gao et al., 2021a; Lin et al., 2022). The multi-vector dense models could also be viewed under this category, where the token representation vectors jointly contribute to the relevancy computation along with the `[CLS]` vector (Khattab and Zaharia, 2020; Li et al., 2022).

3 Methods

In this section, we introduce monoBERT, the reranker we used in this work, and how we apply late interaction on the reranker.

3.1 monoBERT

monoBERT (Nogueira and Cho, 2019) is one of the first works that applied pretrained transformers in passage retrieval. The model is fed with concatenated query q and document d and computes relevance scores $s_{q,d}$ from the `[CLS]` representation on the last layer of the Transformer encoder. We borrow the following formulations from Lin et al. (2020) and Pradeep et al. (2022):

$$s_m(q,d) = T_{\texttt{[CLS]}} W + b \tag{1}$$

where $T_{\texttt{[CLS]}} \in \mathbf{R}^D$ is the `[CLS]` representation on the final layer, and $W \in \mathbf{R}^{D\times 1}$ and $b \in \mathbf{R}$ are the weight and bias for classification.

The model may also use other pretrained Transformers checkpoints following the standard parlance in the community, such as ELECTRA (Clark et al., 2020) which is often referred to as mono-ELECTRA. To avoid confusion with other cross encoders that do not follow this naming scheme, we always refer to the model as monoBERT, and specify the backbone that it is initialized from.

3.2 Late Interaction

In this work, we use the simplest version of late interaction proposed by Khattab and Zaharia (2020). We first obtain the representation of each token in the query q and document d:

$$v_{q_i} = T_{q_i} W_{q_i} + b_{q_i}; \quad v_{d_j} = T_{d_j} W_{d_j} + b_{d_j}; \tag{2}$$

where q_i and d_j represent the i-th token of query q and the j-th token of document d, respectively. Similar to Eq. 1, $T \in \mathbf{R}^D$ is the representation of each token on the final layer. $W \in \mathbf{R}^{D\times D_{tok}}$ and $b \in \mathbf{R}^{D_{tok}}$ are the weight and bias in a projection layer, which projects the T_{tok} to a lower dimension $D_{tok} < D$.

Eq. 2 is the default setting in all experiments. In Section 5.3, we investigate another variant where the token representations are not projected.

With token representations v_{q_i} and v_{d_j}, the late interaction then computes the token interaction scores score by summing up the inner product between all tokens in queries and documents:

$$s_l(q,d) = \sum_{q_i} \max_{d_j} (v_{q_i}^T v_{d_j}) \tag{3}$$

It shares the same formulation with the first-stage retrievers, and only differ in that the token representation T_{q_i} and T_{d_j} are computed jointly with both query and document information, whereas in retrievers, they are computed independently from each other, with T_{q_i} perceiving no information from document d and vice versa.

At training time, we compute the losses $\mathcal{L}_m$ and $\mathcal{L}_l$ based on s_m and s_l (or s'_l), respectively:

$$\mathcal{L} = \mathcal{L}_m(q, d^+, d_1^-, ..., d_n^-) + \mathcal{L}_l(q, d^+, d_1^-, ..., d_n^-)$$

where d^+ is the positive document and $\{d_i^-\}_{i=1}^n$ are the negative documents to the query q. We use LCE (Gao et al., 2021b; Pradeep et al., 2022) for both losses in this work. At inference time, we sum the two scores as the final relevance score, i.e., $s_{final} = s_m + s_l$.[1]

[1]We explored adding weighting terms for s_m and s_c, but

Backbone	Add LI?	MS MARCO MRR@10	BEIR Avg nDCG@10
MiniLM	×	0.390	0.467
	✓	0.392	0.491
ELECTRA$_{\text{base}}$	×	0.400	0.481
	✓	0.402	0.494
ELECTRA$_{\text{large}}$	×	0.413	0.507
	✓	0.413	0.524

Table 2: MRR@10 on MS MARCO and the averaged nDCG@10 scores on BEIR. Rerankers are initialized from MiniLM, ELECTRA$_{\text{base}}$, and ELECTRA$_{\text{large}}$. Results on BEIR rerank the top-1k passages from BM25. **LI**: late-interaction. *Scores on each BEIR dataset are reported in Appendix Table 5 due to space limitation.

Add LI?	*Sparse*			*Multi-vector Dense*
	BM25	**uniCOIL**	**SPLADE**	**ColBERT v2**
×	0.467	0.426	0.469	0.467
✓	0.491	0.452	0.492	0.493

Add LI?	*Single-vector Dense*				
	DPR (Wiki)	**DPR (MS)**	**ANCE**	**TCT**	**TAS-B**
×	0.451	0.474	0.471	0.47	0.472
✓	0.472	0.495	0.493	0.494	0.494

Table 3: Averaged nDCG@10 scores on BEIR, reranking the top-1k passages from each first-stage retriever. **TCT**: TCT-ColBERT. **LI**: late-interaction. *Scores on each BEIR dataset are reported in Appendix Table 6 due to space limitation.

4 Experimental Setup

In this work, we follow the exact same training procedure as Pradeep et al. (2022), detailed in Appendix A. For evaluation, we use MS MARCO (Bajaj et al., 2018) for the in-domain evaluation and 13 datasets from BEIR (Thakur et al., 2021) for OOD evaluation, chosen due to license reasons. BEIR covers 10 domains including Wikipedia, Finance, Scientific, Quora, etc.

At the inference stage, we always rerank top-$1k$ results from the first-stage retrievers. On MS MARCO, we use TCT-ColBERT (Lin et al., 2021b) as the retriever following Pradeep et al. (2022). On BEIR, we use a list of retrievers that covers the categories of sparse, single- and multi-vector dense retrievers. Retrievers results are produced using one of Pyserini (Lin et al., 2021a), BEIR (Thakur et al., 2021) repository, and ColBERT (Khattab and Zaharia, 2020) repository.[2] More details are provided along with the code release.

We experimented with three backbones in this work: MiniLM (Wang et al., 2020), ELECTRA$_{\text{base}}$, and ELECTRA$_{\text{large}}$ (Clark et al., 2020). All models are available on HuggingFace (Wolf et al., 2020).

5 Results and Analysis

Table 1 provides a preview of the effect of adding late interaction on top of rerankers, where we observed it brings a free gain on OOD capacity. In this section, we examine our findings in multiple settings, showing its consistency over different model sizes and first-stage retrievers of different natures.

5.1 Model Size

Previous papers found that the generalization ability could depend on the model scale. Specifically, models with a more extensive set of parameters can better generate on unseen distribution (Ni et al., 2021). This leads to our question: *does late interaction remain helpful in improving OOD capacity when initialized from larger backbone models?*

Surprisingly, the contribution of late interaction barely depends on the model size. Table 2 shows both in-domain (on MS MARCO) and OOD (on BEIR) scores on rerankers without and with adding late interaction after the final layer, where the rerankers are initialized from three different sizes of backbones: MiniLM, ELECTRA$_{\text{base}}$ and ELECTRA$_{\text{large}}$. The size of ELECTRA$_{\text{base}}$/ELECTRA$_{\text{large}}$ is roughly $3.3\times$/$10.3\times$ of MiniLM.[3]

While we observe higher average scores on BEIR as the model size increases, which echoes the previous finding that better generalization ability could be gained as the model scales up. The relative improvement brought by token information is similar across the backbones. On both ELECTRA$_{\text{base}}$ and ELECTRA$_{\text{large}}$, adding late interaction drastically improves the average nDCG@10 on BEIR, from 0.474 to 0.502 with ELECTRA$_{\text{base}}$ and from 0.507 to 0.524 with ELECTRA$_{\text{large}}$. Additionally, the in-domain scores on the other two backbones are not affected as well, suggesting that the "free" gain is consistent over different model sizes.

5.2 First-stage Retriever

We categorize first-stage retrievers into dense and sparse retrievers, and the dense retrievers can be further categorized into single- and multi-vector

only observed marginal gains. Thus we report the simplest formulation here.

[2] https://github.com/stanford-futuredata/ColBERT

[3] MiniLM, ELECTRA$_{\text{base}}$, and ELECTRA$_{\text{large}}$ have 33M, 110M, and 340M parameters respectively.

	Projected Token Dimension (D_{tok})	**MS MARCO** MRR@10	**BEIR** nDCG@10
(1)	$D_{\text{tok}} = 1$	0.3920	0.4890
(2)	$D_{\text{tok}} = 32$	0.3920	0.4914
(3)	$D_{\text{tok}} = 128$	0.3920	0.4910
(4)	$D_{\text{tok}} = 384$	0.3900	0.4911

Table 4: MRR@10 of MS MARCO and nDCG@10 of BEIR, using different dimensions of token representation (D_{tok} in Eq. 2). We report scores to 4 digits here as the values are close in all conditions.

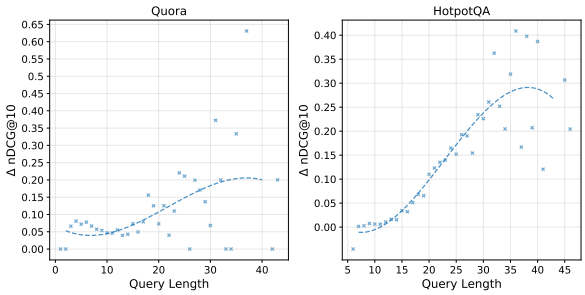

Figure 1: nDCG@10 improvement from late interaction on queries over different lengths. Each point represents the average of nDCG@10 improvements over the query of the corresponding length. The line is the least square polynomial fit of the points.

dense retrievers. We want to explore whether late interaction gives higher improvement when reranking the results of any specific type of retrievers.

Table 3 shows the reranking results on BEIR on an extensive list of retrievers, covering all three categories above. Looking at the averaged nDCG@10 on BEIR, we do not observe any clear preference for any specific category of retrievers. That is, we found that late interaction consistently improves the OOD capacity when using retrievers in different natures, bringing a similar degree of improvement of 0.02-0.03 on average.

5.3 Token Dimensions

In first-stage retrievals, it is common to project the token representation into lower dimensions as restricted by indexing storage space and search efficiency. However, the representations are computed on the fly for rerankers without any index storage. That is, in the context of rerankers, using representations in higher dimensions does not bring any additional storage cost and only minor searching latency. We thus examine whether using higher token dimensions D_{tok} is helpful.

Results are shown in Table 4, where row (2) corresponds to the BM25 results reported in Table 3. Comparing rows (1–4), we found that the token dimensions have little impact on effectiveness on BEIR: on row (1), using $dim = 1$ already obtains 0.4890 on average BEIR, while increasing the dimension to $dim = 32$ and onwards only provides marginal improvement.

5.4 Query Length

Finally, we present our analysis on how the late interaction improves the OOD capacity of rerankers. We explored many aspects of query properties on finding the groups of queries that benefit the most from the late interaction, including the challenging level of the query, the completeness of the positive document, the ranking similarity among different retrievers, and so on. In the end, we found that query length is the most prominent indicator of the per-query improvement.

Figure 1 plots the distribution of nDCG@10 improvement by late interaction according to the query length on Quora and HotpotQA, two datasets included in BEIR.[4] Specifically, each point represents the average of nDCG@10 improvements over the query of the same length (same coordinate on the x-axis). We additionally plot an approximated polynomial line based on the points to better reveal the relationship between the query length and nDCG@10 improvement.

On both datasets, we observe a clear tendency that late interaction brings higher improvement on longer queries. Here we report results using BM25 as the retriever, while the observation is similar when reranking candidates from other retrievers.

6 Conclusion

In this work, we presented our finding that adding late interaction to existing rerankers brings visible improvement to out-of-distribution capacity without any degradation on in-domain effectiveness, even though the reranker already processes the token interaction via the attention mechanism at previous layers. Extensive experiments on different model sizes and first-stage retrievers show that this improvement is consistent, and according to our analysis, the improvement is more prominent on longer queries. Our findings suggest that boiling all information into the `[CLS]` token may not be the optimal choice for neural rerankers, and more studies are required to better explore its capacity.

[4] Length determined as the number of query tokens delimited by whitespace.

7 Limitations

While adding late interaction on top of the cross-encoder provides consistent free gains on the out-of-distribution scenario, one limitation of the work is that the architecture of late interaction is not novel, but rather borrowed from the multi-vector retrieval models (Khattab and Zaharia, 2020). However, we chose to start with this architecture, since it is the simplest form of the between query–document tokens interaction while also achieving top-tier OOD results among the retrievers, especially when adding distillation from cross-encoder and hard negatives training (Santhanam et al., 2022; Formal et al., 2021; Li et al., 2022; Chen et al., 2021). We believe that a simple and effective architecture facilitates its usage by other research works and thus better benefits the community.

References

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2018. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. *arXiv preprint arXiv:1611.09268v3*.

Xilun Chen, Kushal Lakhotia, Barlas Oğuz, Anchit Gupta, Patrick Lewis, Stan Peshterliev, Yashar Mehdad, Sonal Gupta, and Wen-tau Yih. 2021. Salient phrase aware dense retrieval: Can a dense retriever imitate a sparse one? *arXiv preprint arXiv:2110.06918*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2021. Splade v2: Sparse lexical and expansion model for information retrieval.

Luyu Gao, Zhuyun Dai, and Jamie Callan. 2021a. COIL: Revisit exact lexical match in information retrieval with contextualized inverted list. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online. Association for Computational Linguistics.

Luyu Gao, Zhuyun Dai, and Jamie Callan. 2021b. Rethink training of BERT rerankers in multi-stage retrieval pipeline. In *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28 – April 1, 2021, Proceedings, Part II*, page 280–286, Berlin, Heidelberg. Springer-Verlag.

Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16, New York, NY, USA. Association for Computing Machinery.

Kai Hui, Andrew Yates, Klaus Berberich, and Gerard de Melo. 2017. PACRR: A position-aware neural IR model for relevance matching. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark. Association for Computational Linguistics.

Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of the 43rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2020)*, pages 39–48.

Minghan Li, Sheng-Chieh Lin, Barlas Oguz, Asish Ghoshal, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, and Xilun Chen. 2022. Citadel: Conditional token interaction via dynamic lexical routing for efficient and effective multi-vector retrieval. *arXiv preprint arXiv:2211.10411*.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021a. Pyserini: A Python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*, pages 2356–2362.

Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. 2020. Pretrained transformers for text ranking: BERT and beyond.

Sheng-Chieh Lin, Minghan Li, and Jimmy Lin. 2022. Aggretriever: A simple approach to aggregate textual representation for robust dense passage retrieval. *arXiv preprint arXiv:2208.00511*.

Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. 2021b. In-batch negatives for knowledge distillation with tightly-coupled teachers for dense retrieval. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 163–173, Online. Association for Computational Linguistics.

Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. Cedr: Contextualized embeddings for document ranking. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 1101–1104, New York, NY, USA. Association for Computing Machinery.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernández Ábrego, Ji Ma, Vincent Y Zhao, Yi Luan, Keith B Hall, Ming-Wei Chang, et al. 2021. Large dual encoders are generalizable retrievers. *arXiv preprint arXiv:2112.07899*.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with BERT. *arXiv preprint arXiv:1901.04085*.

Ronak Pradeep, Yuqi Liu, Xinyu Zhang, Yilin Li, Andrew Yates, and Jimmy Lin. 2022. Squeezing water from a stone: A bag of tricks for further improving cross-encoder effectiveness for reranking. In *Advances in Information Retrieval*, pages 655–670, Cham. Springer International Publishing.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Seattle, United States. Association for Computational Linguistics.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. MINILM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online.

Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. 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*, SIGIR '17, page 55–64, New York, NY, USA. Association for Computing Machinery.

Andrew Yates, Siddhant Arora, Xinyu Zhang, Wei Yang, Kevin Martin Jose, and Jimmy Lin. 2020a. Capreolus: A toolkit for end-to-end neural ad hoc retrieval. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, WSDM '20, page 861–864, New York, NY, USA. Association for Computing Machinery.

Andrew Yates, Kevin Martin Jose, Xinyu Zhang, and Jimmy Lin. 2020b. Flexible ir pipelines with capreolus. In *Proceedings of the 29th ACM International Conference on Information Knowledge Management*, CIKM '20, page 3181–3188, New York, NY, USA. Association for Computing Machinery.

A Training Configuration

In all experiments, we train the cross-encoders on MS MARCO (Bajaj et al., 2018), a dataset where the queries are prepared from Bing search log, and the collection contains paragraphs from the general Web. It contains 8.8M passages, over 500k query–document pairs for training.

We implement the model based on Capreolus (Yates et al., 2020a,b), an IR toolkit for end-to-end neural ad hoc retrieval that focuses on cross-encoders. We use the logic of training and inference in Capreolus for MS MARCO, and implement the inference on BEIR based on its sample script.[5]

All training configurations follow Pradeep et al. (2022): We train MS MARCO on 30k steps with a learning rate $1e-5$ and batch size 16. We use linear warmup on the first 3k steps, then linearly decay the learning rate on the following steps. Cross-encoders were trained on LCE loss (Gao et al., 2021b; Pradeep et al., 2022) with the number of negative samples to be 7.

All experiments used Quadro RTX 8000 GPUs. It took approximately 8 hours for cross-encoder training and 0.5–24 hours to rerank top-1k documents of BEIR, time depending on the dataset. For each cross-encoder setting, we only performed one training with a fixed seed.

B Results on BEIR

Due to the space limitation, we only report the averaged scores on BEIR in the main paper. In this section, Table 5 and Table 6 presents the full nDCG@10 scores on each BEIR dataset, corresponding to the Table 2 in Section 5.1 (**Model Size**), and Table 3 in Section 5.2 (**First-Stage Retriever**).

C License

The MS MARCO dataset is licensed under Creative Commons Attribution 4.0 International, whereas BEIR datasets and Capreolus toolkit are under Apache License 2.0. The usage of the artifacts in this work is consistent with their intended use. Since our codebase is extended from Capreolus, it would inherit the Apache License 2.0.

[5] https://github.com/beir-cellar/beir

Backbone	Add LI?	MS MARCO (MRR@10)	BEIR (nDCG@10) Avg	TREC-COVID	NF Corpus	NQ	Hotpot QA	FiQA	Argu Ana	Touche-2020	Quora	DB Pedia	SCI DOCS	FEVER	Climate-FEVER	Sci Fact
MiniLM	×	0.390	0.467	0.699	0.355	0.504	0.620	0.359	0.335	0.308	0.722	0.426	0.151	0.754	0.164	0.679
	✓	0.392	0.491	0.705	0.349	0.501	0.673	0.360	0.527	0.324	0.784	0.424	0.155	0.723	0.172	0.691
ELECTRA$_{base}$	×	0.400	0.481	0.727	0.362	0.523	0.660	0.389	0.291	0.317	0.773	0.436	0.152	0.748	0.112	0.669
	✓	0.402	0.494	0.736	0.368	0.527	0.714	0.401	0.443	0.320	0.690	0.449	0.162	0.740	0.152	0.715
ELECTRA$_{large}$	×	0.413	0.507	0.801	0.380	0.559	0.733	0.453	0.250	0.339	0.772	0.468	0.181	0.791	0.149	0.719
	✓	0.413	0.524	0.786	0.378	0.559	0.735	0.457	0.436	0.335	0.800	0.460	0.182	0.769	0.179	0.733

Table 5: MRR@10 on MS MARCO and nDCG@10 scores on BEIR. Rerankers are initialized from MiniLM, ELECTRA$_{base}$, and ELECTRA$_{large}$. Results on BEIR rerank the top-1k passages from BM25. **LI**: late-interaction.

First Stage	Add LI?	BEIR (nDCG@10) Avg	TREC-COVID	NF Corpus	NQ	Hotpot QA	FiQA	Argu Ana	Touche-2020	Quora	DB Pedia	SCI DOCS	FEVER	Climate-FEVER	Sci Fact
Sparse															
BM25	×	0.467	0.699	0.355	0.504	0.620	0.359	0.335	0.308	0.722	0.426	0.151	0.754	0.164	0.679
	✓	0.491	0.705	0.349	0.501	0.673	0.360	0.527	0.324	0.784	0.424	0.155	0.723	0.172	0.691
uniCOIL	×	0.426	0.711	0.337	0.556	0.576	0.271	0.335	0.277	0.727	0.426	0.152	0.375	0.116	0.680
	✓	0.452	0.713	0.328	0.552	0.625	0.272	0.555	0.285	0.784	0.423	0.156	0.360	0.128	0.691
SPLADE	×	0.469	0.706	0.336	0.563	0.617	0.362	0.320	0.278	0.728	0.434	0.152	0.758	0.160	0.682
	✓	0.492	0.699	0.330	0.560	0.671	0.361	0.526	0.288	0.786	0.432	0.157	0.717	0.173	0.691
Single-vector Dense															
DPR (Wiki)	×	0.451	0.699	0.335	0.571	0.600	0.341	0.333	0.285	0.523	0.433	0.154	0.753	0.175	0.662
	✓	0.472	0.715	0.330	0.568	0.643	0.339	0.524	0.296	0.557	0.432	0.156	0.721	0.180	0.673
DPR (MS)	×	0.474	0.737	0.334	0.562	0.613	0.364	0.336	0.278	0.718	0.434	0.153	0.771	0.181	0.677
	✓	0.495	0.738	0.329	0.557	0.655	0.364	0.528	0.287	0.782	0.434	0.156	0.738	0.186	0.687
ANCE	×	0.471	0.724	0.331	0.554	0.594	0.360	0.338	0.285	0.717	0.419	0.155	0.781	0.192	0.676
	✓	0.493	0.740	0.327	0.550	0.626	0.363	0.529	0.291	0.781	0.418	0.157	0.750	0.192	0.687
TCT-ColBERT	×	0.470	0.719	0.336	0.564	0.620	0.360	0.319	0.281	0.714	0.437	0.154	0.767	0.170	0.676
	✓	0.494	0.725	0.330	0.560	0.665	0.360	0.524	0.291	0.780	0.438	0.157	0.733	0.177	0.689
TAS-B	×	0.472	0.714	0.338	0.565	0.623	0.361	0.333	0.281	0.727	0.436	0.153	0.760	0.167	0.680
	✓	0.494	0.713	0.331	0.560	0.670	0.358	0.527	0.292	0.787	0.435	0.157	0.729	0.176	0.689
Multi-vector Dense															
ColBERT v2	×	0.467	0.707	0.333	0.564	0.621	0.360	0.316	0.278	0.716	0.434	0.152	0.756	0.156	0.679
	✓	0.493	0.709	0.327	0.560	0.672	0.361	0.525	0.291	0.780	0.431	0.157	0.724	0.178	0.691

Table 6: nDCG@10 scores on BEIR, reranking the top-1k passages from each first-stage retriever. **LI**: late-interaction.