Evaluation of Models for Ranking of Long Documents (online appendix)

Anonymous Author(s)*†‡§

1 TRAINING SETUP (ADDITIONAL DETAILS)

In our setup, a ranker is applied to the output of the first-stage retrieval model, also known as a candidate-generator. Thus, the ranker is trained to distinguish between positive examples (known relevant documents) and hard negative examples (documents not marked as relevant) sampled from the set of top-k candidates returned by the candidate generator. We used $k = 100$ for MS MARCO datasets and $k = 1000$ for Robust04. Depending on the experiment and the dataset we use different candidate generators: for MS MARCO v1 and Robust04 we used a BM25 ranker [19]. In that, for MS MARCO v1 it was applied to documents expanded using the doc2query approach [14]. For MS MARCO v2, we used a hybrid retriever where candidate records are first produced using a k-NN search and subsequently re-ranked using a linear fusion of BM25 scores and the cosine similarity between query and document embeddings. Embeddings were generated using ANCE [24].

All experiments were carried out using the an **anonymous** retrieval toolkit framework, which employed Lucene and an **anonymous** toolkit for k-NN search to provide retrieval capabilities. Deep learning support was provided via PyTorch [15] and HuggingFace Transformers library [23]. The instructions to reproduce our key results are publicly available.[1] an **anonymous** retrieval toolkit allowed us to use virtually any HuggingFace Transformer [21] model available in the online model repository: When we use such a model for initialization, we discard all prediction heads. In particular, we used BigBird [25] and the Electra [2] model provided via Sentence-BERT library [18] as vanilla BERT rankers.[2] The basic vanilla BERT ranker as well as CEDR models are virtually unmodified implementations from the CEDR framework [12]. We re-implemented PARADE models from scratch: This allowed us to use (almost) any pre-trained model from the Huggingface repository as an aggregator Transformer. Specifically, we chose a six-layer MiniLM [22] cross-encoder from the Sentence-BERT library, which is "prefinetuned" on MS MARCO (we discard the prediction head).[3] For randomly initialized aggregator Transformer we—somewhat similar to [10]—used a BERT model consisting of two layers and four attention heads.

All MS MARCO models were trained from scratch. Then these models were fine-tuned on Robust04. Note that except for the aggregation and interaction Transformers, we use a *base*, i.e., a 12-layer Transformer [21] models. BERT-base is more practical then a 24-layer BERT-large and performs at par with BERT-large on MS MARCO and Robust04 [8, 10]. In our own experiments, we see that large (24 and more layers) model perform much better on the MS MARCO Passage collection, but we were not able to outperform 12-layer models on the MS MARCO Documents collection. Note that Longformer [1], BigBird [25], and DEBERTA base [7], JINA [6], and MOSAIC [16] all have 12 layers, but a larger embedding matrix.

One training epoch consisted in iterating over all queries and sampling one positive and one negative example with a subsequent computation of a pairwise margin loss. We used the minbatch size 1 with gradient accumulation over 16 steps. The learning rates are provided in the model configuration files (in the online appendix). We used the AdamW optimizer [11] and a constant learning rate with a 20% linear warm-up [13].

Moreover, for MS MARCO, all models except PARADE-Transf-Pretr-LATEIR-L6 and PARADE-Transf-RAND-L2 were trained for one epoch: Further training did not improve accuracy and sometimes were deterimental. However, PARADE-Transf-RAND-L2 and PARADE-Transf-Pretr-LATEIR-L6 required two-to-three epochs to reach the maximum accuracy. One epoch on MS MARCO took from 6 hours for BERT FirstP to 24 hours for Longformer (Using NVIDIA RTX 3090 TI). In the case of Robust04, each model was finetuned for 100 epochs.

From our experience models trained on MS MARCO v2 performed worse on TREC 2021 queries compared to models trained on MS MARCO v1. This may indicate that models somehow learn to distinguish between original MS MARCO v1 documents and newly added ones (which did not have positive judgements in the training sets). As a result, these models are biased and tend to not rank these new documents well even when they are considered to be relevant by NIST assessors. For this reason, we used MS MARCO v2 data in a zero-shot transfer mode where ranking models trained on MS MARCO v1 are evaluated on TREC DL 2021 queries.

We have learned that—unlike neural *retrievers*—BERT-base [3] *rankers* are relatively insensitive to learning rates, their schedules, and the choice of loss functions. We were sometimes able to achieve better results using multiple negatives per query and a listwise margin loss (or cross-entropy). However, the gains were small and not consistent compared to a simple pairwise margin loss used in our work (in fact, using a listwise loss function sometimes lead to overfitting). Note again that this is different from neural *retrievers* where training is difficult without using a listwise loss and/or batch-negatives [4, 9, 17, 24, 26].

Except for ablations, each model was trained using *three* seeds using half-precision. MOSAIC models were trained using full-precision. Their training was unstable (even though we used the full precision) and often resulted in close-to-zero performance. For this reason we continued training with MORE seeds until we obtained three models with reasonable performance. This seed selection strategy could potentially have biased (up) MOSAIC results.

To compute statistical significance, we averaged query-specific metric values over these seeds. Queries were padded to 32 BERT

[1] https://anonymous.4open.science/r/long_doc_rank_model_analysis_v2-78E9/

[2] Longformer requires a custom model wrapper, because we need to "place" global attention on query tokens.

[3] cross-encoder/ms-marco-electra-base

tokens (longer queries were truncated). To enable efficient training and evaluation of the large number of models, documents were truncated to have at most 1431 BERT tokens. Thus, long document were split into at most three chunks containing 477 document tokens (each concatenated with up to 32 query tokens plus three special tokens).

We evaluated 20 models, but we had to exclude two LongT5 variants [5] and RoFormer (with ROPE embeddings) [20] due to poor convergence and/or crashes. Specifically, LongT5 models were ≈ 10% less accurate than BERT FirstP after 10 epochs. Given the uncertainty regarding the possible convergence of models as well as the need to train these for three epochs, we have to abandon this experiment as overly expensive. RoFormer models were failing due to CUDA errors when the context length exceeded 512: We were not able to resolve this issue.

2 MS MARCO SYNTHETIC

The MS MARCO synthetic collection was created from the MS MARCO passage collection in such a way that each document contains exactly one relevant passage and this passage does not start before token 512. The length of a document is (nearly) uniformly distributed between 512 and $1431 = (512 - 32 - 3) \times 3$, which corresponds to the case when a greedy disjoint-partitioning approach can process three 512-token chunks, three special tokens, and the query length of at most 32 tokens.

We created one document for each training or development query that had a relevant passage as follows using the below described algorithm. Note that we generated about 7K test and close to 500K training queries, but used only 50K for fine-tuning and 1K for testing. On one hand, this was sufficient for accurate training and testing and, on the other hand, it reduced experimentation time and cost.

Assume that C_t is the number of tokens in the passage:

- Select randomly a document length between $512 + C_t$ and 1431;
- Using rejection sampling, obtain K_1 non-relevant samples such that their length exceeds 512, but the length of $K_1 - 1$ first samples is at most 512.
- Using the same approach, sample another $K_2 + 1$ samples such that the total length of K_2 samples is at most $1431 - C_t$, but the total length of $K_2 + 1$ samples exceeds this value.
- Discard the last sampled passage and randomly mix the last K_2 non-relevant passages with a single relevant passage.
- Finally, append the resulting string to the concatenation of the first K_1 non-relevant passages.

3 DETAILED EXPERIMENTAL RESULTS

Detailed experiments for MS MARCO development, TREC DL, and Robust04 datasets are presented in Table 1 and Table 2 after the bibliography.

REFERENCES

[1] Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The Long-Document Transformer. *CoRR* abs/2004.05150 (2020).

[2] 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*. OpenReview.net.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. (2019), 4171–4186.

[4] Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2021. SPLADE v2: Sparse Lexical and Expansion Model for Information Retrieval. *CoRR* abs/2109.10086 (2021).

[5] Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2022. LongT5: Efficient Text-To-Text Transformer for Long Sequences. In *Findings of the Association for Computational Linguistics: NAACL 2022*, Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz (Eds.). Association for Computational Linguistics, Seattle, United States, 724–736. https://doi.org/10.18653/v1/2022.findings-naacl.55

[6] Michael Günther, Jackmin Ong, Isabelle Mohr, Alaeddine Abdessalem, Tanguy Abel, Mohammad Kalim Akram, Susana Guzman, Georgios Mastrapas, Saba Sturua, Bo Wang, Maximilian Werk, Nan Wang, and Han Xiao. 2023. Jina Embeddings 2: 8192-Token General-Purpose Text Embeddings for Long Documents. arXiv:2310.19923 [cs.CL]

[7] Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing. arXiv:2111.09543 [cs.CL]

[8] Sebastian Hofstätter, Markus Zlabinger, and Allan Hanbury. 2020. Interpretable & Time-Budget-Constrained Contextualization for Re-Ranking. In *ECAI (Frontiers in Artificial Intelligence and Applications, Vol. 325)*. IOS Press, 513–520.

[9] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *EMNLP (1)*. Association for Computational Linguistics, 6769–6781.

[10] Canjia Li, Andrew Yates, Sean MacAvaney, Ben He, and Yingfei Sun. 2024. PARADE: Passage Representation Aggregation for Document Reranking. *ACM Trans. Inf. Syst.* 42, 2 (2024), 36:1–36:26. https://doi.org/10.1145/3600088

[11] Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101* (2017).

[12] Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. CEDR: Contextualized Embeddings for Document Ranking. In *SIGIR*. ACM, 1101–1104.

[13] Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2020. On the Stability of Fine-tuning BERT: Misconceptions, Explanations, and Strong Baselines. *CoRR* abs/2006.04884 (2020).

[14] Rodrigo Nogueira and Jimmy Lin. 2019. From doc2query to docTTTTTquery. https://cs.uwaterloo.ca/~jimmylin/publications/Nogueira_Lin_2019_docTTTTTquery-latest.pdf [Last Checked Apr 2022]. *MS MARCO passage retrieval task publication* (2019).

[15] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*. 8026–8037.

[16] Jacob Portes, Alexander R Trott, Sam Havens, DANIEL KING, Abhinav Venigalla, Moin Nadeem, Nikhil Sardana, Daya Khudia, and Jonathan Frankle. 2023. MosaicBERT: A Bidirectional Encoder Optimized for Fast Pretraining. In *Thirty-seventh Conference on Neural Information Processing Systems*. https://openreview.net/forum?id=5zipcfLC2Z

[17] Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2021. RocketQA: An Optimized Training Approach to Dense Passage Retrieval for Open-Domain Question Answering. In *NAACL-HLT*. Association for Computational Linguistics, 5835–5847.

[18] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 3982–3992.

[19] Stephen Robertson. 2004. Understanding inverse document frequency: on theoretical arguments for IDF. *Journal of Documentation* 60, 5 (2004), 503–520.

[20] Jianlin Su, Murtadha H. M. Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. RoFormer: Enhanced transformer with Rotary Position Embedding. *Neurocomputing* 568 (2024), 127063.

[21] 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 *NIPS*. 5998–6008.

[22] 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 *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 5776–5788.

[23] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi 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 M. Rush. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *ArXiv* abs/1910.03771 (2019).

Table 1: Ranking Performance on MS MARCO and TREC DL.

Model	MS MARCO dev	TREC DL 2019-2021		
	MRR	NDCG@10	P@10	MAP
FirstP (BERT)	0.394	0.632	0.712	0.311
FirstP (Longformer)	0.404	0.643	0.722	0.317
FirstP (ELECTRA)	0.417	0.662	0.734	0.320
FirstP (DEBERTA)	0.415	0.672	0.741	0.327
FirstP (Big-Bird)	0.408	0.656	0.727	0.321
FirstP (JINA)	0.422	0.654	0.728	0.320
FirstP (MOSAIC)	0.423	0.643	0.726	0.316
AvgP	0.389 (−1.3%)	0.642 (+1.5%)	0.717 (+0.7%)	0.317^a (+2.0%)
MaxP	0.392 (−0.4%)	0.644^a (+1.9%)	0.723 (+1.5%)	0.322^a (+3.7%)
MaxP (ELECTRA)	0.414 (−0.6%)	0.659 (−0.5%)	0.745 (+1.5%)	0.326 (+2.1%)
SumP	0.390 (−1.0%)	0.639 (+1.0%)	0.715 (+0.4%)	0.319^a (+2.6%)
CEDR-DRMM	0.385^a (−2.3%)	0.629 (−0.5%)	0.708 (−0.5%)	0.313 (+0.6%)
CEDR-KNRM	0.379^a (−3.8%)	0.630 (−0.3%)	0.711 (−0.1%)	0.313 (+0.8%)
CEDR-PACRR	0.395 (+0.3%)	0.643^a (+1.6%)	0.719 (+0.9%)	0.320^a (+2.9%)
Neural Model1	0.398 (+0.9%)	0.650^a (+2.8%)	0.723^a (+1.5%)	0.323^a (+3.9%)
PARADE Attn	0.416^a (+5.5%)	0.652^a (+3.1%)	0.728^a (+2.2%)	0.324^a (+4.2%)
PARADE Attn (ELECTRA)	0.431^a (+3.3%)	0.680^a (+2.7%)	0.763^a (+3.9%)	0.335^a (+4.9%)
PARADE Attn (DEBERTA)	0.422^a (+1.6%)	**0.688**^{a} (+2.4%)	**0.763**^{a} (+3.0%)	**0.339**^{a} (+3.9%)
PARADE Avg	0.392 (−0.6%)	0.646^a (+2.1%)	0.715 (+0.4%)	0.317^a (+2.1%)
PARADE Max	0.405^a (+2.7%)	0.655^a (+3.5%)	0.733^a (+2.9%)	0.324^a (+4.1%)
PARADE Transf-RAND-L2	0.419^a (+6.3%)	0.655^a (+3.6%)	0.734^a (+3.1%)	0.326^a (+5.0%)
PARADE Transf-RAND-L2 (ELECTRA)	**0.433**^{a} (+3.9%)	0.670 (+1.2%)	0.747 (+1.8%)	0.327 (+2.2%)
PARADE Transf-PRETR-L6	0.402^a (+1.9%)	0.643 (+1.6%)	0.717 (+0.8%)	0.322^a (+3.6%)
PARADE Transf-PRETR-LATEIR-L6	0.398 (+1.1%)	0.626 (−0.9%)	0.707 (−0.7%)	0.307 (−1.1%)
LongP (Longformer)	0.412^a (+1.9%)	0.668^a (+3.9%)	0.752^a (+4.1%)	0.333^a (+5.1%)
LongP (Big-Bird)	0.397^a (−2.9%)	0.651 (−0.7%)	0.726 (−0.2%)	0.322 (+0.3%)
LongP (JINA)	0.416^a (−1.5%)	0.665^a (+1.7%)	0.742^a (+2.0%)	0.328^a (+2.4%)
LongP (MOSAIC)	0.421 (−0.4%)	0.664^a (+3.3%)	0.740^a (+1.9%)	0.327^a (+3.7%)

In each column we show a relative gain with respect model's respective *FirstP* baseline: The last column shows the average relative gain of *FirstP*. Best numbers are in **bold**: Results are averaged over three seeds. Unless specified explicitly, the backbone is **BERT-base**.
Statistical significant differences with respect to this baseline are denoted using the superscript superscript **a**. p-value threshold is 0.01 for an MS MARCO development collection and 0.05 otherwise.

[24] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. In *ICLR*. OpenReview.net.

[25] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big Bird: Transformers for Longer Sequences. In *NeurIPS*.

[26] George Zerveas, Navid Rekabsaz, Daniel Cohen, and Carsten Eickhoff. 2021. CODER: An efficient framework for improving retrieval through COntextualized Document Embedding Reranking. *ArXiv* abs/2112.08766 (2021).

Table 2: Ranking Performance on Robust04.

Model	NDCG@20	P@20	MAP	NDCG@20	P@20	MAP
FirstP (BERT)	0.475	0.405	0.277	0.527	0.447	0.303
FirstP (Longformer)	0.483	0.413	0.277	0.540	0.454	0.307
FirstP (ELECTRA)	0.492	0.424	0.294	0.552	0.465	0.320
FirstP (DEBERTA)	0.534	0.459	0.319	0.596	0.503	0.350
FirstP (Big-Bird)	0.507	0.435	0.300	0.560	0.473	0.325
FirstP (JINA)	0.488	0.421	0.287	0.532	0.450	0.305
FirstP (MOSAIC)	0.453	0.390	0.266	0.538	0.455	0.310
AvgP	0.478 (+0.5%)	0.411 (+1.6%)	0.292^{a} (+5.4%)	0.531 (+0.9%)	0.451 (+1.0%)	0.324^{a} (+6.7%)
MaxP	0.488^{a} (+2.6%)	0.425^{a} (+5.1%)	0.306^{a} (+10.6%)	0.544^{a} (+3.3%)	0.467^{a} (+4.5%)	0.338^{a} (+11.5%)
MaxP (ELECTRA)	0.502 (+2.0%)	0.441^{a} (+3.9%)	0.319^{a} (+8.3%)	0.563 (+2.1%)	0.483^{a} (+4.0%)	0.350^{a} (+9.3%)
SumP	0.486 (+2.2%)	0.418^{a} (+3.4%)	0.305^{a} (+10.2%)	0.538 (+2.1%)	0.461^{a} (+3.1%)	0.337^{a} (+11.1%)
CEDR-DRMM	0.466 (−2.0%)	0.403 (−0.4%)	0.287^{a} (+3.8%)	0.533 (+1.3%)	0.458^{a} (+2.5%)	0.326^{a} (+7.6%)
CEDR-KNRM	0.483 (+1.7%)	0.413 (+1.9%)	0.291^{a} (+5.1%)	0.535 (+1.7%)	0.456 (+2.0%)	0.324^{a} (+6.8%)
CEDR-PACRR	0.496^{a} (+4.3%)	0.426^{a} (+5.3%)	0.307^{a} (+11.0%)	0.549^{a} (+4.2%)	0.466^{a} (+4.4%)	0.337^{a} (+11.2%)
Neural Model1	0.484 (+1.8%)	0.417^{a} (+3.1%)	0.298^{a} (+7.7%)	0.537 (+1.9%)	0.459^{a} (+2.6%)	0.330^{a} (+8.8%)
PARADE Attn	0.503^{a} (+5.7%)	0.433^{a} (+6.9%)	0.311^{a} (+12.4%)	0.556^{a} (+5.6%)	0.476^{a} (+6.5%)	0.344^{a} (+13.3%)
PARADE Attn (ELECTRA)	0.523^{a} (+6.4%)	0.456^{a} (+7.4%)	0.329^{a} (+11.7%)	0.581^{a} (+5.3%)	0.495^{a} (+6.5%)	0.358^{a} (+11.9%)
PARADE Attn (DEBERTA)	**0.549**^{a} (+2.9%)	**0.475**^{a} (+3.6%)	**0.346**^{a} (+8.7%)	**0.615**^{a} (+3.2%)	**0.522**^{a} (+3.8%)	**0.383**^{a} (+9.4%)
PARADE Avg	0.483 (+1.5%)	0.412 (+1.8%)	0.291^{a} (+5.2%)	0.534 (+1.5%)	0.457 (+2.4%)	0.318^{a} (+4.7%)
PARADE Max	0.489^{a} (+2.8%)	0.420^{a} (+3.8%)	0.306^{a} (+10.8%)	0.548^{a} (+4.0%)	0.470^{a} (+5.3%)	0.337^{a} (+11.0%)
PARADE Transf-RAND-L2	0.488^{a} (+2.8%)	0.423^{a} (+4.6%)	0.303^{a} (+9.7%)	0.548^{a} (+4.1%)	0.469^{a} (+5.0%)	0.338^{a} (+11.6%)
PARADE Transf-RAND-L2 (ELECTRA)	0.523^{a} (+6.3%)	0.454^{a} (+6.9%)	0.330^{a} (+12.2%)	0.574^{a} (+3.9%)	0.488^{a} (+5.0%)	0.354^{a} (+10.6%)
PARADE Transf-PRETR-L6	0.494^{a} (+4.0%)	0.426^{a} (+5.3%)	0.308^{a} (+11.5%)	0.554^{a} (+5.1%)	0.474^{a} (+6.1%)	0.346^{a} (+14.1%)
PARADE Transf-PRETR-LATEIR-L6	0.450^{a} (−5.2%)	0.389^{a} (−3.9%)	0.277 (+0.3%)	0.501^{a} (−4.9%)	0.423^{a} (−5.3%)	0.302 (−0.5%)
LongP (Longformer)	0.500^{a} (+3.6%)	0.435^{a} (+5.3%)	0.309^{a} (+11.5%)	0.568^{a} (+5.1%)	0.482^{a} (+6.1%)	0.347^{a} (+12.9%)
LongP (Big-Bird)	0.452^{a} (−10.9%)	0.389^{a} (−10.7%)	0.274^{a} (−8.8%)	0.477^{a} (−14.9%)	0.400^{a} (−15.5%)	0.279^{a} (−14.2%)
LongP (JINA)	0.503^{a} (+2.9%)	0.434^{a} (+3.1%)	0.309^{a} (+7.5%)	0.558^{a} (+4.9%)	0.473^{a} (+5.2%)	0.335^{a} (+9.7%)
LongP (MOSAIC)	0.456 (+0.6%)	0.393 (+0.8%)	0.280^{a} (+5.3%)	0.570^{a} (+6.0%)	0.484^{a} (+6.3%)	0.350^{a} (+13.0%)

In each column we show a relative gain with respect model's respective *FirstP* baseline: The last column shows the average relative gain of *FirstP*. Best numbers are in **bold**: Results are averaged over three seeds. Unless specified explicitly, the backbone is **BERT-base**.
Statistical significant differences with respect to this baseline are denoted using the superscript superscript **a**. p-value threshold is 0.05.