Learned Token Pruning in Contextualized Late Interaction over BERT (ColBERT)

Carlos Lassance
Naver Labs Europe
Meylan, France
first.lastatnaverlabs.com

Maroua Maachou
Naver Labs Europe
Meylan, France
first.last-internatnaverlabs.com

Joohee Park
Naver
Seoul, Korea
james.first.lastatnavercorp.com

Stéphane Clinchant
Naver Labs Europe
Meylan, France
first.lastatnaverlabs.com

ABSTRACT

BERT-based rankers have been shown very effective as rerankers in information retrieval tasks. In order to extend these models to full-ranking scenarios, the ColBERT model has been recently proposed, which adopts a late interaction mechanism. This mechanism allows for the representation of documents to be precomputed in advance. However, the late-interaction mechanism leads to large index size, as one needs to save a representation for each token of every document. In this work, we focus on token pruning techniques in order to mitigate this problem. We test four methods, ranging from simpler ones to the use of a single layer of attention mechanism to select the tokens to keep at indexing time. Our experiments show that for the MS MARCO-passages collection, indexes can be pruned up to 70% of their original size, without a significant drop in performance. We also evaluate on the MS MARCO-documents collection and the BEIR benchmark, which reveals some challenges for the proposed mechanism.

CCS CONCEPTS

• **Information systems** → **Information retrieval**.

KEYWORDS

information retrieval, token pruning, BERT, ColBERT

ACM Reference Format:
Carlos Lassance, Maroua Maachou, Joohee Park, and Stéphane Clinchant. 2022. Learned Token Pruning in Contextualized Late Interaction over BERT (ColBERT). In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22), July 11–15, 2022, Madrid, Spain.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3477495.3531835

SIGIR '22, July 11–15, 2022, Madrid, Spain

ACM ISBN 978-1-4503-8732-3/22/07…$15.00
https://doi.org/10.1145/3477495.3531835

1 INTRODUCTION

In recent Information Retrieval (IR) systems, Pretrained Language Models (PLM) [11] have taken the state of the art by storm. Two main families of PLM retrieval methods have been developed:

(1) Representation-based [13], where both query and documents are encoded separately into a single representation and scoring is performed via distance between representations;
(2) Interaction-based [12], where a query-document pair is treated jointly by a neural network to generate a score.

On one hand, the former is very efficient as representations of documents can be indexed and only the query has to be computed during inference time. On the other hand, the latter has better performance as it is able to perform a more thorough scoring between queries and documents. In order to bridge the gap between these two families, the ColBERT [9] method indexes a representation per token, which allows to precompute document representations and part of the capability of an interaction model (each contextualized token of the query interacts with each precomputed contextualized token of the document). However, the ColBERT advantage comes with an important cost on the index size, since every token (rather than a pooled version of a document) needs to be indexed.

In this work we investigate ColBERT models, by looking into two characteristics of the representations: normalization and query expansion. We then focus on the index size by limiting the amount of tokens to be saved in each document using 4 methods based on i) position of the token, ii) Inverse Document Frequency (IDF) of the token, iii) special tokens, and iv) attention mechanism of the last layer. To empirically evaluate our method, we perform our investigation under the most common benchmark for neural IR (MS MARCO on both passage and document tasks), showing that we are able to greatly improve efficiency (in terms of index size and complexity) while still maintaining acceptable effectiveness (in terms of MRR and Recall).

2 RELATED WORK

Efficient Pretrained LMs: In NLP, there has been a lot of work seeking to improve the efficiency of pretrained LMs. For instance, quantization and distillation have been extensively studied in this context [14]. Closest to our work, Power-BERT [6] and length adaptive transformers [10] have been proposed to reduce the number of FLOPS operations, by eliminating tokens in the PLM layers.

Index pruning in IR:. Pruning indexes is a traditional method in Information Retrieval to reduce latency and memory requirements and has been studied thoroughly in many contributions (e.g [21] and [2]), as far as we are aware this is the first application to PLM token-level indexing.

Improving ColBERT Efficiency. One way to mitigate the problem of large index size of ColBERT is to reduce the dimensionality and apply quantization to the document tokens. Note that this is already done by default as the output of the PLM (most of the time of 768 dimensions) is projected into a smaller space (128 dimensions). In the original paper, the authors show very good performance by both reducing the dimensionality and quantizing the tokens [9]. In this sense, a binarization technique [20] has been proposed for information retrieval, and concurrently with this work, experiments have shown that it is indeed possible to binarize ColBERT tokens [15].

The present work is orthogonal to the previously presented research direction, in that we aim to reduce the index size by removing tokens from the index, instead of reducing their size. We consider the combination of those research directions as necessary to improve ColBERT models, but leave it as future work. Lastly, ColBERT has been extended to perform pseudo relevance feedback in [19] and query pruning has been studied to improve latency in [17].

3 METHODOLOGY AND COLBERT

The ColBERT model is based on a transformer [3] encoder for documents and queries. Each item Q or D is encoded into a representation $\mathbf{R}$ which is a matrix of dimensions (t, d) where t is the amount of tokens in the item and d is the encoding dimension of each token. The score of a document (D) for a given query (Q) is given by the following formula:

$$s(D,Q) = \sum_{j=0}^{|t_Q|} \max_{i=0}^{|t_D|} (\mathbf{R}_D \cdot \mathbf{R}_Q)_{i,j} \ . \tag{1}$$

This late-interaction model improves computation efficiency compared to a full-interaction model, because representations $\mathbf{R}_D$ for all documents ($D \in \mathcal{D}$) can be precomputed. In this way, during inference only the query representation needs to be computed. However, this incurs a very large representation size by document (dt) which can quickly become intractable when the amount of documents increases. As those are the only factors impacting index size, there are two solutions: i) reduce the amount of dimensions per token, and ii) reduce the amount of tokens per document. While many solutions exist for the first case, we are not aware of any previous work for the second one.

Normalization and query expansion: In an effort to ensure that the scores of each query and document are not dependent on their length, each token is normalized in the l2-hyper-sphere. Also queries are expanded so that they have the same length (32 tokens). We show via experiments that this is not actually needed for all PLM backbones (c.f. Section 4), which reduces the inference cost significantly.

3.1 Limiting the amount of tokens

We investigate four diverse methods for limiting the amount of tokens that are stored for each document. These methods are integrated **during the training** of our ColBERT models[1]. In other words, we add a pooling layer on top of ColBERT to select a maximum of k tokens per document and then use the ColBERT scoring equation restricted to the k selected tokens.

***First** k tokens:* One very simple, but also very strong, baseline is to keep only the k first tokens of a document. Indeed such a baseline takes advantage of the inherent bias of the MS MARCO dataset where the first tokens are considered to be the most important [8]. This is the only method where we do not remove punctuation.

*Top k **IDF** tokens:* Another possibility is to choose the rarest tokens of a document, in other words, the ones with the highest Inverse Document Frequency (IDF). This should allow us to keep only the most defining words of a document.

*k-**Unused Tokens (UNU).*** We also test the possibility of adding k special tokens ('unused' terms from the BERT vocabulary) to the start of the document and keep only those tokens. The kept tokens are always the same for all documents and always in the same positions, which increases the consistency for the model, which we posit leads to easier optimization. However this approach has some drawbacks: i) increased computation time as it forces documents to have at least k tokens, and ii) possibly missing context from long documents as truncation is done with less "real-word" tokens.

Attention score (ATT):. We propose to use the *last layer of attention of the PLM* as a way to detect the most important $k < t$ tokens of the document. Recall that the attention of a document (A_D) is a three-dimensional tensor (h, t, t'), where the first dimension is the number of attention heads (number of different "attentions") and the last two dimensions represent the amount of attention a token t "pays" to token t', which is normalized so that for each token t the sum of attentions t' is 1. We propose an importance score per token which is the sum of attention payed to the token over all heads:

$$i(t)_D = \sum_{i=0}^{h} \sum_{j=0}^{|t_D|} (A_D)_{i,t,j} \tag{2}$$

So that at the end we only keep $k < t$ tokens, based on the top-k scores. Note that this score is computed over the *last layer of the document encoding*, and not in the interaction between documents and queries. In other words it is *independent* from queries.

4 EXPERIMENTS

Experimental setup: In this work, we train our ColBERT [9] models using the MINILM-12-384 [18][2] . Models are trained during 150k steps, with a linear learning rate from 0 to the final learning rate (2e-5) warmup of 10k steps and linear learning rate annealing (2e-5 to 0) for the remaining 140k steps. We consider both **the passage and the document task** and use the original triplet files provided by

[1] We tested both during and just after training and noticed that integrating this during training improved the results.
[2] available at https://huggingface.co/microsoft/MiniLM-L12-H384-uncased

the MS MARCO organizers [1]. We evaluate our models in the full ranking scenario, **using a brute force implementation**, which differs from the original ColBERT paper, which uses approximate nearest neighbors for that scenario. By not using ANN search, we avoid the need of ANN-hyperparameter tuning and/or the risk of unfair comparisons. Nevertheless, we have experimented with ANN and results are very close to the ones presented in the initial paragraphs. Finally, we train and evaluate "passage" models with a max length of 180 tokens (truncating longer sequences), while "document" models have a max length of 512.

Investigating ColBERT modelling: First, we investigate the default ColBERT model and verify if the normalization and query expansion operations are helpful for all PLM. Results for this ablation are presented in Table 1. We observe that when using MINILM instead of BERT there is no need for normalization and expansion. Therefore we can skip these steps which reduces latency (search complexity depends linearly to the amount of tokens in the query). For the rest of this work, we only use models without these operations.

Table 1: Ablation of token normalization and query expansion on the MINILM ColBERT model.

Normalization	Expansion	MRR@10	Recall@1000
X	X	0.362	96.3%
X		0.055	41.2%
	X	0.363	97.0%
		0.365	**97.1%**

Reducing index size on passage dataset: We now evaluate the ability of the 4 proposed token pruning methods (c.f. Section 3.1). Results are displayed in Table 2. We notice that when keeping the top $k = 50$, almost all the tested methods allow us to reduce the amount of tokens by 30% while keeping similar acceptable performance (less than 0.01 MRR@10 drop, less than 1% Recall@1000 drop). On the other hand, when we drop to $k = 10$ there is a noticeable drop in MRR@10 performance for all methods, with *Unused Tokens* getting the best results, which shows that this technique is very useful for small k but not for larger ones. Note that further tests in other datasets could be useful to verify not only this difference between *Unused Tokens* and the other methods, but also of the *First* method that uses the inherent bias of MS MARCO to the start of the document [8]. Note that % of tokens may vary due to: the size of passages (*k=50*); the use or not of punctuation (*First*); repeated scores (*attention* and *IDF*) or because a method increases the original passage length (*Unused Tokens*).

Reducing index size on document dataset: In the case of the document task, the mean amount of tokens per document is higher than in the case of passages. Due to this disparity we test only the case of $k = 50$, which allows us to cut 85% of the document tokens. While $k = 50$ provides a noticeable reduction of index size, it still means that the amount of data to keep for every document is large ($50d$). For example, while the document($k = 50$) relative reduction of index size is the same as passage with $k = 10$, the final index size is still twice as large. Lastly, as it was the case with the smaller token size on the passage task, both *Unused Tokens* and *First* present the best results.

Table 2: Results for different methods and k tokens to keep on MS MARCO. We use MRR@10 for the passage dataset and MRR@100 for the document one. Index size consider 2kb per vector (128 dimensions in fp16).

Method	% tokens	Size (Gb)	MRR	Recall@1000
Baseline	100%	142	0.365	97.1%
Passage (k=50)				
First	72.6%	103	0.357	96.7%
IDF	71.9%	102	0.355	96.7%
Unused Tokens	74.1%	101	0.342	96.2%
Attention Score	71.1%	105	**0.358**	96.7%
Passage (k=10)				
First	14.8%	21	0.302	93.3%
IDF	15.3%	22	0.290	91.0%
Unused Tokens	14.8%	21	**0.314**	**94.0%**
Attention Score	14.8%	21	0.281	91.9%
Document (k=50)				
Baseline	100%	290	0.380	95.6%
First	13.1%	38	0.347	91.6%
IDF	13.5%	39	0.225	81.2%
Unused Tokens	13.1%	38	**0.354**	**93.1%**
Attention Score	13.1%	38	0.306	90.9%

Results on the BEIR benchmark. We now verify how do the networks learned with token pruning generalize outside the MS MARCO domain. To do so, we use the BEIR benchmark [16], using the subset of the 13 datasets that are ready to use[3]. The models we compare to are the standard ColBERT [9], using the numbers reported in [16], our baseline ColBERT, that uses the MINILM PLM and does not perform neither expansion or normalization and the 4 models trained for passage retrieval with $k = 50$ (i.e. max sequence length of 180 tokens). The results are displayed in Table 3.

First, we note that all the models trained for this work outperform the original ColBERT [9] on Arguana, which creates some noise on the results. We believe that this comes from the fact that Arguana queries are actually documents and so [9] underperforms due to its query expansion, but we did not delve deeper to validate this claim. Second, we note a slightly larger loss of performance to the one we had previously seen in MS MARCO (around 6% mean nDCG@10 loss on BEIR, compared to 2% MRR@10 on MS MARCO), this comes mostly because the size of the documents is increased for the BEIR datasets (i.e. more compression). However, this is not the only (and maybe not even the major) component of the performance loss, as datasets with similar size to MS MARCO (for example HotpotQA) also present steep losses of performance. Finally, there is a larger gain of performance for the attention method compared to the

[3]The other 5 datasets either have a license that makes it harder to obtain or are not evaluated in the same way as the other 13

Table 3: NDCG@10 results on BEIR. Bold values are the best in the row for its category (baseline or pruned).

Dataset	Colbert Baselines [9, 16]	Ours	Top-50 pruned, trained on passage First	IDF	UNU	ATT
ArguAna	0.233	**0.446**	0.421	0.424	0.410	**0.443**
Climate-FEVER	**0.184**	0.168	0.140	0.140	0.123	**0.157**
DBPedia	0.392	**0.404**	0.375	**0.383**	0.372	0.382
FEVER	**0.771**	0.734	**0.660**	0.613	0.638	0.654
FiQA-2018	0.317	**0.326**	0.265	0.286	0.274	**0.306**
HotpotQA	0.593	**0.631**	0.533	0.556	0.515	**0.573**
NFCorpus	0.305	**0.319**	0.286	0.259	0.285	**0.288**
NQ	**0.524**	0.507	0.475	0.442	0.462	**0.490**
Quora	**0.854**	0.850	0.850	0.844	0.807	**0.852**
SCIDOCS	0.145	**0.150**	0.135	0.130	**0.138**	0.136
SciFact	**0.671**	0.633	0.545	0.475	0.502	**0.564**
TREC-COVID	0.677	**0.715**	**0.691**	0.600	0.665	0.678
Touché-2020	0.202	**0.230**	**0.232**	0.209	0.228	0.203
Avg. zero-shot	0.451	**0.470**	0.431	0.412	0.417	**0.440**

Table 4: NDCG@10 results on BEIR. Bold values are the best in the row for the pruned models

Dataset	ColBERT Document Baseline	Top-50 pruned, trained on document First	IDF	UNU	ATT
ArguAna	0,436	0,407	0,434	0,407	**0,443**
Climate-FEVER	0,206	0,152	0,110	**0,158**	0,133
DBPedia	0,405	0,377	**0,387**	0,363	0,373
FEVER	0,810	0,655	0,475	**0,694**	0,631
FiQA-2018	0,348	0,279	0,271	0,295	**0,305**
HotpotQA	0,635	0,533	0,554	0,520	**0,555**
NFCorpus	0,330	0,286	0,230	**0,290**	0,281
NQ	0,509	0,472	0,438	0,466	**0,472**
Quora	0,851	0,846	0,847	0,802	**0,848**
SCIDOCS	0,152	0,129	0,122	0,131	**0,136**
SciFact	0,680	0,533	0,436	**0,569**	0,542
TREC-COVID	0,709	**0,700**	0,561	0,671	0,607
Touché-2020	0,230	0,222	0,171	**0,234**	0,209
Avg. zero-shot	0,485	0,430	0,387	**0,431**	0,426

others than we had seen on MS MARCO, which could possibly show that outside of the inherent bias of MS MARCO for the start of passages [8], such methods could indeed thrive.

We also tested the models trained for document retrieval and display the results in Table 4. Compared to passage retrieval, the baseline achieves a better mean performance, while some of the pruned models present reduced average performance (Attention Score and IDF), others gain performance (Unused Token) or keep the same performance (First). The loss of performance of Attention and IDF seems to come from the fact that with more tokens these models tend to repeat words (c.f. Section 5), while on the other hand the gain of the Unused Tokens seems to come from the fact that as the networks can treat more tokens, there is less loss of information from increasing the sequence size (so less of the real sequence is thresholded out of the input). Lastly, it is not surprising that First keeps the same performance as it keeps the same tokens independently of its training and/or max sequence length.

Results using ANN. For completeness, we also run retrieval of passages following the approximate nearest neighbor scheme described in the original ColBERT paper [9] and available at: https://github.com/stanford-futuredata/ColBERT/. For the first stage token retrieval we use a probing of the 128 closests clusters and perform full retrieval on all documents that had tokens in the top 8k of each query token. Results are available in Table 5.

Table 5: Results for different methods and k tokens to keep on MS MARCO-passage using ANN retrieval

Method	% tokens	MRR@10	Recall@1000
Baseline	100%	0.365	96.5%
k=50			
First	72.6%	0.357	96.0%
IDF	71.9%	0.355	96.0%
Attention Score	71.1%	**0.358**	**96.2%**
k=10			
First	14.8%	**0.302**	**93.2%**
IDF	15.3%	0.290	90.9%
Attention Score	14.8%	0.281	92.1%

Notably, the difference in MRR@10 is negligeable as none of the 7 networks we evaluate shows a difference of more than 0.001 MRR@10. On the other hand, when we look into Recall@1000, differences start to appear as the baseline loses 0.6%, the $k = 50$ models lose in average 0.7%. However for the $k = 10$ models there is no such loss of performance. This confirms that while results using ANN are very close to those of brute-force, they could have impacted the results, especially if we had applied the same ANN to the datasets from BEIR as the number of tokens (and so the size of clusters) varies from dataset to dataset.

5 RESULT ANALYSIS

Analysis of indexed documents: We analyzed the tokens selected by the attention mechanism on the document set. One problem we observed is about repetitions: it selects of the same token at different position. For instance, we show two examples below of the selected tokens of two different documents to demonstrate the repetition and the stemming effect (dog vs dogs, cake vs ##cake):

['[CLS]', 'dog', 'nasal', 'poly', 'removal', 'dog', 'nasal', 'poly', 'removal', 'poly', 'grow', 'body', 'parts', 'dogs', 'nasal', 'poly', 'dog', 'being', 'dog', 'nasal', 'poly', 'dog', 'dog', 'nasal', 'poly', 'removal', 'nasal', 'poly', 'dogs', 'poly', 'dog', 'dogs', 'nasal', 'poly', 'dogs', 'hide', '", 'dogs', 'dog', 'nasal', 'growth', 'nasal', 'poly', 'dogs', '", 'dogs', 'nose', 'dogs', 'nose', '[SEP]']

['[CLS]', 'ba', 'mini', 'cup', '##cake', 'cake', 'mix', 'ba', 'mini', 'cup', '##cake', 'cake', 'mix', 'cup', '##cake', 'sweet', 'ba', 'cake', 'mini', 'cup', '##cake', 'cup', 'cake', 'mini', 'cup', '##cake', 'oven', 'mini', 'cup', '##cake', 'pan', 'mini', 'cake', 'mini', 'cup', '##cake', 'mini', 'cup', 'mini', 'cup', '##cake', 'cup', '##cake', 'cup', '##cake', 'cup', '##cake', 'mini', 'cup', '[SEP]']

These results indicates that token selection methods should take into account either the repetition problem or model the diversity of the selected tokens, which we leave for future work.

ColBERT relevance: Note that while ColBERT was the state of the art for MS MARCO full ranking during its release it has been overtook by dense [5, 7] and sparse [4] siamese models that have better training procedure, namely: i) hard-negative mining [4, 5, 7], ii) distillation [4, 7], and iii) pretraining [5]. When this work was conducted, these techniques were not yet tested for the ColBERT model, so we did not apply them, but concurrently to this work they have been shown beneficial in a pre-print [15]. Finally, ColBERT has also demonstrated interesting properties such as better zero-shot performance [16] and being able to combine with traditional IR techniques such as PRF [19].

6 CONCLUSION

In this work we investigate the ColBERT model and test different methods to reduce its late-interaction complexity and its problematic index size. We first verify that for some PLM (namely MINILM) we do not need to perform normalization of tokens and query augmentation, which thus improve latency. Furthermore, we have also demonstrated that some very simple methods allow to remove 30% of the tokens from the passage index without incurring in any consequent performance loss. On the other hand the MS MARCO-document collection reveals challenges for such mechanism, where even pruning 90% of tokens may not be enough reduction. The combination of such token pruning methods with already studied embedding compression methods could lead to further improvement of ColBERT-based IR systems.

REFERENCES

[1] 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:1611.09268 [cs.CL]

[2] David Carmel, Doron Cohen, Ronald Fagin, Eitan Farchi, Michael Herscovici, Yoelle S Maarek, and Aya Soffer. 2001. Static index pruning for information retrieval systems. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. 43–50.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805

[4] Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2021. SPLADE v2: Sparse Lexical and Expansion Model for Information Retrieval. arXiv:2109.10086 [cs.IR]

[5] Luyu Gao and Jamie Callan. 2021. Unsupervised Corpus Aware Language Model Pre-training for Dense Passage Retrieval. arXiv:2108.05540 [cs.IR]

[6] Saurabh Goyal, Anamitra Roy Choudhury, Saurabh Raje, Venkatesan Chakaravarthy, Yogish Sabharwal, and Ashish Verma. 2020. PoWER-BERT: Accelerating BERT Inference via Progressive Word-vector Elimination. In *Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119)*, Hal Daumé III and Aarti Singh (Eds.). PMLR, 3690–3699. https://proceedings.mlr.press/v119/goyal20a.html

[7] Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan Hanbury. 2021. Efficiently Teaching an Effective Dense Retriever with Balanced Topic Aware Sampling. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 113–122. https://doi.org/10.1145/3404835.3462891

[8] Sebastian Hofstätter, Aldo Lipani, Sophia Althammer, Markus Zlabinger, and Allan Hanbury. 2021. Mitigating the Position Bias of Transformer Models in Passage Re-Ranking. *arXiv preprint arXiv:2101.06980* (2021).

[9] Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 39–48. https://doi.org/10.1145/3397271.3401075

[10] Gyuwan Kim and Kyunghyun Cho. 2021. Length-Adaptive Transformer: Train Once with Length Drop, Use Anytime with Search. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Online, 6501–6511. https://doi.org/10.18653/v1/2021.acl-long.508

[11] Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. 2020. Pretrained Transformers for Text Ranking: BERT and Beyond. *arXiv:2010.06467 [cs]* (Oct. 2020). http://arxiv.org/abs/2010.06467 ZSCC: NoCitationData[s0] arXiv: 2010.06467.

[12] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. arXiv:1901.04085 [cs.IR]

[13] Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084* (2019).

[14] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108* (2019).

[15] Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2021. ColBERTv2: Effective and Efficient Retrieval via Lightweight Late Interaction. arXiv:2112.01488 [cs.IR]

[16] 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)*. https://openreview.net/forum?id=wCu6T5xFjeJ

[17] Nicola Tonellotto and Craig Macdonald. 2021. Query Embedding Pruning for Dense Retrieval. *CoRR* abs/2108.10341 (2021). arXiv:2108.10341 https://arxiv.org/abs/2108.10341

[18] 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. arXiv:2002.10957 [cs.CL]

[19] Xiao Wang, Craig Macdonald, Nicola Tonellotto, and Iadh Ounis. 2021. Pseudo-Relevance Feedback for Multiple Representation Dense Retrieval. In *ICTIR '21*, Faegheh Hasibi, Yi Fang, and Akiko Aizawa (Eds.). ACM, 297–306. https://doi.org/10.1145/3471158.3472250

[20] Ikuya Yamada, Akari Asai, and Hannaneh Hajishirzi. 2021. Efficient Passage Retrieval with Hashing for Open-domain Question Answering. arXiv:2106.00882 [cs.CL]

[21] Justin Zobel and Alistair Moffat. 2006. Inverted files for text search engines. *ACM computing surveys (CSUR)* 38, 2 (2006), 6–es.