An Experimental Study on Pretraining Transformers from Scratch for IR

Carlos Lassance, Hervé Dejean, and Stéphane Clinchant

Naver Labs Europe, Meylan, France
{carlos.lassance,herve.dejean,stephane.clinchant}@naverlabs.com

Abstract. Finetuning Pretrained Language Models (PLM) for IR has been de facto the standard practice since their breakthrough effectiveness few years ago. But, is this approach well understood? In this paper, we study the impact of the pretraining collection on the final IR effectiveness. In particular, we challenge the current hypothesis that PLM shall be trained on a large enough generic collection and we show that pretraining from scratch on the collection of interest is surprisingly competitive with the current approach. We benchmark first-stage ranking rankers and cross-encoders for reranking on the task of general passage retrieval on MSMARCO, Mr-Tydi for Arabic, Japanese and Russian, and TripClick for specific domain. Contrary to popular belief, we show that, for finetuning first-stage rankers, models pretrained solely on their collection have equivalent or better effectiveness compared to more general models. However, there is a slight effectiveness drop for rerankers pretrained only on the target collection. Overall, our study sheds a new light on the role of the pretraining collection and should make our community ponder on building specialized models by pretraining from scratch. Last but not least, doing so could enable better control of efficiency, data bias and replicability, which are key research questions for the IR community.

Keywords: Pretrained Language Models · Transformers · IR

1 Introduction

Transformers models are the main breakthrough in artificial intelligence over the past five years. Pretraining transformers models with Masked Language Modeling (MLM), a form of self-supervision, as proposed in the seminal BERT model [11] led to major improvement in natural language processing, computer vision and other domains. Pretraining and self-supervision have then paved the way to a race on bigger foundation models. Information Retrieval (IR) followed the same trajectory, where Pretrained Language Models (PLM) have largely outperform previous neural model [13,15,30,23] but also traditional bag-of-words approaches such as BM25 [46]. These advances were all made possible by a combination of large datasets, PLMs such as BERT, but also priors coming from traditional IR methods such as BM25.

However, these PLMs do not perform well out-of-the box for Information Retrieval (on the contrary to NLP [1]) as they require a significant fine-tuning procedure. In IR, there are two types of PLM: the cross-encoders for reranking a set of top k documents and the dual encoders to deal with an efficient first-stage retrieval [30]. However, the standard pretraining MLM task may not be the best task for IR as argued in [14]. More-so, a growing tendency is to introduce a "middle-training" step [14] to bridge this gap and adapt the PLM not only to the retrieval domain, but also to the way the sentences will be encoded. For example [15,14,34] adapt the Masked Language Modeling (MLM) loss, so that the model learns to condense information on the CLS token, which will be used as the de-facto sentence embedding during fine-tuning. Similarly, several middle training tasks have been proposed to better fit the IR tasks or by using weak supervision.

On the one hand, it seems that there is a widespread belief that the downstream effectiveness is essentially due to pretraining on a *large* external collection. For instance, the foundation models report [4] state that "AI is undergoing a paradigm shift with the rise of models (e.g., BERT, DALL-E, GPT-3) that are trained on broad data at scale and are adaptable to a wide range of downstream tasks." On the other hand, the middle training process seems to contradict the former. This is why, in this paper, we aim to investigate the following questions: do we actually need large scale pretraining for Neural IR? How much knowledge is actually encoded from the large pretraining collection? Besides, what is known about pretraining in IR is largely limited by the MSMARCO setting and therefore a related question is how one shall address pretraining language models for new languages or new domains when it comes to IR.

In this paper, we aim to verify if these preconceived notions are needed, or if we could just have combined the pretraining and middle training steps to generate PLMs that are already adapted to the problem at hand with a smaller cost than doing both separately. Overall, this paper makes the following contributions:

1. We study pretrained transformers from scratch on IR collections;
2. We show that first-stage rankers, pretrained on MSMARCO, are as effective or even better in-domain (MSMARCO), while out-of-domain those models generalize as well (sparse retrieval) or worse (dense retrieval);
3. We evaluate cross encoders that are pretrained from scratch and verify that they actually benefit from external pretraining;
4. We show that first-stage retrievers, trained from scratch on the target collection, are competitive or outperform domain specific models (e.g. SciBERT on TripClick) and multilingual models (e.g. MContriever on Mr. TiDy);
5. Variants of Transformers architectures, such as DeBERTa, alleged to be better in NLP benchmarks do not bring benefits to IR, even when trained from scratch.

[1] For instance, freezing the BERT encoding and learning an additional linear layer is sufficient to obtain good performance in NLP [11], while such approach is not as effective in IR.

2 Related Work

PLMs in IR: today, the standard practice of many IR researchers is to simply download an existing pretrained model in order to finetune it on their retrieval task. After their success on reranking, PLMs have been adopted for first-stage ranking with a bi-encoder network to tackle efficiency requirements [25,30,43] or with late interaction [26]. Several training strategies have been proposed to improve the effectiveness of bi-encoders, such as distillation [20,31,22,48] and hard negative mining [53,42,45]. Parallel to these developments, another research direction aimed at learning *sparse* representations from PLM to behave as lexical matching. COIL [16] (later improved in uniCOIL [29]) learns term-level dense representations to perform contextualized lexical match. SPLADE [13,28] directly learns high-dimensional sparse representation thanks to the MLM head of the PLM and the help of sparse regularization. Most notably, SPLADE achieved state-of-the art effectiveness on the zero-shot benchmark BEIR [51], being later surpassed by other methods with much more compute [36].

Rise of middle training: several works recently proposed to perform an additional step of pretraining, before the final finetuning stage, a procedure that we will call here *middle training*. The rationale is that the PLM weights or its CLS pooling mechanism are not well-suited for retrieval or similarity tasks often used in IR. Two main ideas emerge from this literature: i) using a contrastive loss on different document spans, and ii) using an information bottleneck to better pre-condition the network to rely on its CLS representation to perform predictions [27]. In [6], the paper compares the Inverse Cloze Task, Wiki Link Prediction and Body First Selection. Their result show that a combination of all these tasks was beneficial compared to MLM pretraining only. In [35], hyperlinks are used for pretraining. Another pretraining relies on web page structure and their DOM in the WebFormer model [18]. Furthermore, Contriever [23] relies on contrastive loss from different text spans, similarly to Co-Condenser [15]. Co-Condenser extends the Condenser [14] idea which focuses on middle training the CLS token. Very recently, Retro-MAE [32] revisits the same idea, by masking twice an input passage so that the first masking produce a CLS representation reused for decoding the second masking of the passage. Pretraining for sparse models has been recently investigated in [28] to better condition the network with SPLADE finetuning. The idea is to reuse the FLOPS [41] regularization used during finetuning within the MLM middle training. In [1], 14 different pretraining tasks are compared when training BERT models, including predicting the *tfidf* scores of a document, which was shown to be beneficial. They then evaluate on several NLP tasks, including sentence similarity. In addition, [34,33] propose pretraining with representative word predictions: for each document a set of important word is defined by several heuristics and the model is pretrained to predict that set of words.

While pretraining for IR seems very trendy, the idea of pretraining representation for IR tasks can actually be traced before the advent of PLM (or Foundation Models). For instance, more than ten years ago, the supervised se-

mantic indexing model [2], used hyperlinks anchor to build triplets in a contrastive task, which can be viewed as an ancestor to the pretrainings tasks on Wikipedia. Similarly, weak supervision coming from BM25 was used to pretrain neural IR models [10] before the use of PLM.

Foundational models and architectures: a loosely related line of work is the scaling laws literature for large pretrained language models [24]. The scaling law aims to understand how the architecture and model size influence the perplexity and accuracy of the language model. In [50], Tay et al. showed that perplexity was a poor predictive measure of downstream effectiveness and propose to favor depth rather than just width (i.e. hidden size) in a pattern they named *deep-narrow* architectures. While this question could be interesting to our work, most of the literature is focusing on the large data and model regime, while in IR we look to the other side of the spectrum with small or moderate size collections/models for efficiency purposes.

Finally, a very interesting work by Tay et al. [49] argued that pre-training and architectural advances have been conflated. In addition, they show that convolutional models are in fact competitive with transformers when they are pre-trained on the same collection and for tasks which do not require cross attention between two sentences (e.g a bi-encoder network). Finally, they argued that the current approach is misguided and that both architecture and pretraining should be considered independently. Finally, [7] studies the impact of the pretraining collection for machine translation and in [12] for image related tasks with large models. Our work is related to those, as we study the impact of the pretraining collection for the final effectiveness of an IR system.

3 Pretraining from Scratch

All in all, the role of finetuning representation or performing a middle training seems important for IR systems. The PLM representations seem critical but for good effectiveness, the impact of the pretraining collection on the final effectiveness is unclear since these representations are then finetuned by diverse means. Is the effectiveness heavily influenced by the pretraining collection and its co-occurrence statistics? To investigate this question, we experiment with several PLM models **trained from scratch** on the target collection. This way, we will be able to measure the effectiveness gains obtained by pretraining on a larger external collection and answer the following questions: is there an advantage in pretraining directly and only on the target collection? When and what are the advantages of pretraining on a different larger collection?

On the one hand, pretraining from scratch on the target collection could have the advantage of better modelling the target collection by having more informative co-occurrence statistics between tokens. Moreover, 'smaller' sized models may be able to reach the same level of effectiveness, i.e. one can also include efficiency requirements when training these models from scratch. On the other hand, pretraining on a larger collection may lead to more robust/generic as

the model has seen more domains, different token usages and could have 'more' knowledge.

Therefore, by comparing these two approaches, we hope to better understand how large external pretraining contributes to the final effectiveness. In this paper, our research question deals with general-purpose vs specific-purpose model. The mainstream approach is to adopt the general purpose model, by simply adapting it to an IR task. On contrary, this paper investigates specific purpose models to assess their effectiveness (cf. Figure 1). In a nutshell, would pretraining from scratch work for IR models? More specifically, we look at the following research questions:

1. Do we need an external pretrained language model for Information Retrieval?
2. Do models pretrained on target collections generalize to other domains and tasks?
3. Can we take advantage of pretraining for specialized domains and non-English languages?
4. Does efficient pretraining allow us to use recent architectural advances of transformers?

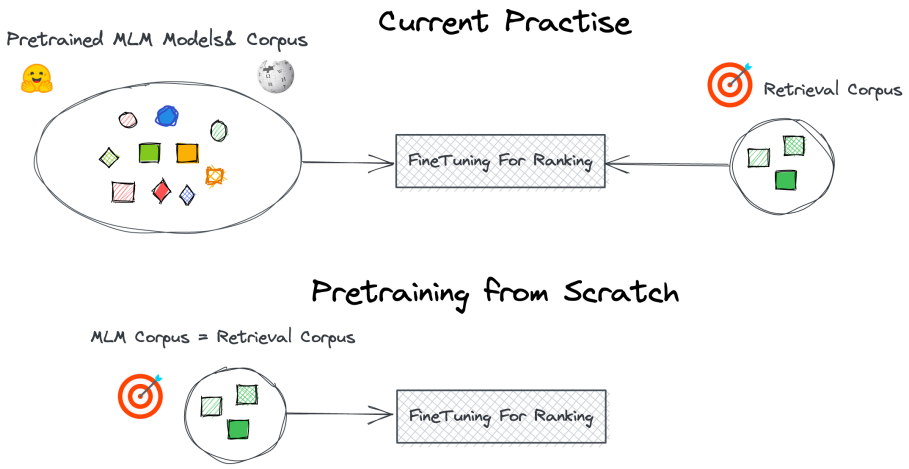

Fig. 1. Pretraining from Scratch: MLM is only performed on the retrieval collection.

To answer these questions, we compare the performance of standards models, such as BERT pretrained in Wikipedia and Book Corpus, to pretraining for scratch on the retrieval collections. For instance, we use the classical MS-MARCO [39] dataset. By pretraining only on MSMARCO, we observe if there is indeed a benefit when pretraining on Wikipedia. Furthermore, we assess whether such models still generalise well on the BEIR dataset [51]. Indeed, one advantage of pretraining on a large collection could be better generalization in zero shot

Table 1. Model Comparison.

	BERT	DistilBERT	MLM 6L	MLM 12L
MLM Collection	Wikipedia, BookCorpus	Wikipedia, BookCorpus	MSMARCO or Mr. TyDi or TripClick	
Pretraining Size (words)	3,300M	3,300M	From 100M to 500M	
# params	110M	66M	67M	110M
# Layers	12	6	6	12
# number_heads	12	16	16	12

settings. For specialized domains, such as health and biomedical data, we use the TripClick [44] dataset and compare pretraining from scratch to the performance of SciBERT [3] and PubMedBERT [17]. Similarly, by pretraining models solely on TripClick, we compare the performance against models trained on a much larger collection (i.e PubMed). Finally, we will extend the previous experiments on non-English datasets, on Mr. TyDi [54] dataset by comparing to MContriever [23], which relied on a pretraining on large dataset with many languages. By conducting those experiments, we can assess whether an external pretrained language model is needed for Information Retrieval and if pretraining from scratch is an interesting alternative.

We will focus our study on BERT [11] and DistilBERT [47] architectures since they are the most popular ones in IR. Furthermore, we will measure the effectiveness of different architecture: *dense* bi-encoders ([43,30]), *SPLADE*, a state of the art sparse model [13,28] and cross-encoders for reranking [40].

3.1 Pretraining

In the following we always pretrain at least two types of models from scratch: a model with 12 layers based on the BERT architecture [11] and one with 6 layers based on DistilBERT [47]. We use BERT and DistilBERT as baselines all along the article. Table 1 summarises the main model characteristics we will consider. For pretraining from scratch, we always fix the vocabulary size to 32k (slightly larger than BERT's 30.5k), using wordpiece [52] to find the most common tokens of the target collection. We refer to those models as MLM 12L and MLM 6L.

We also use 2 models built on a setting called, MLM+FLOPS 12L and MLM+FLOPS 6L, by adding the FLOPS regularization [41], which helps to (pre)condition PLM for usage with SPLADE as proposed in [28]. The standard MLM loss is modified as follows: the MLM logits go through the SPLADE activation function (i.e. $\log(1 + ReLU(y_{\text{logits}}))$), which defines an MLM loss over a sparse set of logits. Finally, another term (FLOPS regularization) is added to force the logits to not only be nonnegative, but actually be sparse. As in SPLADE, a max pooling of the overall sentence is done to get a representation at the word level. On this final representation, the FLOPS regularization forces sparsification (and uniformity) over the overall vocabulary. The total loss is given by $\ell_{\text{MLM}} + \ell_{\text{MLM-SPLADE}} + \ell_{\text{FLOPS}}$.

Pretaining time is between 6 hours to 1 day depending on the models and collections on 8 NVIDIA A100 80 Gb, compared to the original computational

cost of BERT (3 days using 64 TPUs), and the cost of finetuning (around 1 day on 4 V100 32Gb), we consider our pretraining cost to be reasonable.

4 Experiments

Our research question is to assess whether models fully trained on the target collection perform as well as the generic DistilBERT or BERT models. First, we check the results on the general collection MSMARCO [39] (RQ1) and how the models trained on it generalize (RQ2) to a zero-shot scenario in BEIR [51]. We then verify if the results generalize to more specialized collections and non-English languages (RQ3). Finally, we take advantage of the fact that we are training models from scratch to test variants of transformer architectures (RQ4). Additional finetuning details are available at the end of the respective sections.

Experimental Setup Pretraining is performed using 8 NVIDIAs A100 80Gb either on MS-MARCO (RQ1 and RQ2), or TripClick (RQ3), or Mr. TyDi dataset (RQ3). We always use a learning rate of 1e-4. In the case of the MSMARCO collection, we pretrain using MLM and MLM+FLOPS using the entire passage collection (8.8M) combined with the training queries ($\tilde{8}$00k) for a total of (9.6M "documents"), while for TripClick we separate the documents into training and validation (90/10 split). For all collections, the batch size per GPU is either 150 (12L) or 200 (6L). We use an exponential warmup for FLOPS of 5k steps, a warmup of 1k steps for the logits and a learning rate warmup of 10k steps. The λ factor of FLOPS is set to $1e-3$, the max length before truncation to 256 tokens and the networks are pretrained for 125k steps. For Mr. TyDi, we pretrain 3 networks (Arabic, Russian, Japanese) using MLM+FLOPS on the entire language's passage collection (2M for Arabic, 7M for Japanese and 9.6M for Russian) with a batch size per GPU of 200. Finally for TripClick, the only difference is the number of epochs: 60 epochs, and the batch size per GPU is 256 for the 6L models, and 128 for the 12L models.

For finetuning, we used 4 V100 32gb for a sparse model (SPLADE) and a dense neural bi-encoder model. Please note that **we do not use distillation** from a reranker as it would entail transferring information from an existing PLM. For SPLADE, we use the L1 regularization over queries (following [28]) with $\lambda_q = 1e-3$, and for documents a FLOPS regularisation with $\lambda_d = 1e-3$ ($\lambda_d = 5e-4$ on TripClick) following [28]. The learning rate is set up to $2e{-}5$. In all of our tables, **superscripts** denote **significant differences** according to a paired Student's t-test with Bonferroni's correction and $p \leq 0.05$ with the corresponding table row; MRR@10 and nDCG@10 have been multiplied by 100.

4.1 RQ1: Are models fully trained on MSMARCO as good as models pretrained on a diverse collection set?

The first question we want to address is whether models that are solely trained on MSMARCO are as good as models that have used external corpora for pre-

Table 2. Comparison on MSMARCO of first stage sparse neural (SPLADE) models. † indicates a method pretrained solely on MSMARCO.

#	Pretrained Model	R-FLOPS	MSMARCO dev		TREC DL 19		TREC DL 20	
			MRR@10	R@1k	nDCG@10	R@1k	nDCG@10	R@1k
a	Distilbert	1.10	0.373	0.975^{b}	**0.732**	**0.853**	0.708	0.867^{b}
b	BERT	1.32	0.367	0.968	0.727	0.832	0.699	0.842
c	MLM 6L†	8.2	0.370	$\mathbf{0.982}^{ab}$	0.701	0.847	0.700	**0.877**
d	MLM+Flops 6L†	0.72	$\mathbf{0.382}^{abc}$	0.979^{b}	0.698	0.836	0.701	0.872
e	MLM+Flops 12L†	0.97	0.379^{bc}	0.980^{ab}	0.709	0.835	**0.709**	0.865

training (i.e. BookCorpus and Wikipedia). We first investigate models trained as first stage rankers, either using a dense bi-encoder [30,25] or sa SPLADE model.

We finetuned the models using negatives that were sampled from a previously trained SPLADE model. For each V100 we use a batch of the maximum size that does not exceed the total memory. Batches are constructed such that "one element" of the batch is composed of the query itself, a positive passage related to the query and 16 or 32 negatives, depending on the network size (the actual batch sizes per GPU varies from 54 to 102 depending on the pretrained model size). Finetuning is considered finished after two epochs (arbitrary decision based on initial experiments with a validation set). Note that the finetuning setting for both first stage and cross-encoders are almost the same, except for the fact that cross-encoders do not use in-batch negatives and use a learning rate of 1e-4.

We report the retrieval flops (noted R-FLOPS) for SPLADE models, i.e., the number of floating point operations on the inverted index to return the list of documents for a given query. The R-FLOPS metric is defined by an estimation of the average number of floating-point operations between a query and a document which is defined as the expectation $\mathbb{E}_{q,d}\left[\sum_{j \in V} p_j^{(q)} p_j^{(d)}\right]$ where p_j is the activation probability for token j in a document d or a query q. It is empirically estimated from a set of approximately 100k development queries, on the MS MARCO collection. It is thus an indication of the inverted index sparsity and of the computational retrieval cost for a sparse retriever (which is different from the inference cost of the model).

First stage retrievers results are described in Table 2 for sparse models and Table 3 for dense models. Surprisingly, models trained solely on MSMARCO with MLM+FLOPS actually **can perform statistically significantly better** than their counterparts pretrained over larger collections, in both sparse and dense scenarios, while there's no statistically significant difference when considering just MLM on MSMARCO vs MLM on larger corpora. This shows that not only pretraining does not seem to care for a diverse collection, but that by focusing only on "off-the-shelf models" we could be losing possible performance gains of better initialized models. Also note that less computing was used to pretrain MLM+FLOPS 6L compared to its DistilBERT counterpart.

Table 3. Comparison on MSMARCO of first stage dense neural (DPR) models. † indicates a method pretrained solely on MSMARCO.

#	Pretrained Model	MSMARCO dev		TREC DL 19		TREC DL 20	
		MRR@10	R@1k	nDCG@10	R@1k	nDCG@10	R@1k
a	Distilbert	0.342	0.961	0.673	0.774	0.670	0.816
b	BERT	0.347	0.961	**0.697**	0.785	**0.682**	0.809
c	MLM 6L†	0.346	0.968^{ab}	0.664	0.783	0.657	0.818
d	MLM+FLOPS 6L †	0.349	0.968^{ab}	0.670	0.781	0.668	0.837
e	MLM+FLOPS 12L†	$\mathbf{0.352}^{a}$	$\mathbf{0.969}^{ab}$	0.672	**0.800**	0.680	$\mathbf{0.848}^{bc}$

Table 4. Comparison of rerankers on MSMARCO. Models with † were pretrained solely on MSMARCO.

#	Pretrained Model	MSMARCO dev **MRR@10**	TREC DL 2019 **nDCG@10**	TREC DL 2020 **nDCG@10**
a	Without reranking (First stage)	0.384	0.718	**0.737**
b	Distilbert	0.396^{af}	**0.764**	0.734
c	Bert	$\mathbf{0.404}^{abf}$	0.750	0.737
d	MLM 6L †	0.398^{af}	0.743	0.716
e	MLM+Flops 6L †	0.396^{af}	0.724	0.736
f	MLM+Flops 12L †	0.381	0.730	0.722

Finally, we also test in the case of reranking using cross-encoders. Results are available in Table 4. Overall there's no statistical significant gain on the models pretrained with external collections.

4.2 RQ2: Do models pretrained in MSMARCO generalize well on other collections?

In the previous section, we considered only in-domain results. While they have shown that we can outperform (or at least keep comparable effectiveness) using solely the target collection, they could be masking a possible gap in out-of-domain data. In order to verify that models solely pretrained and fine-tuned on MSMARCO do not lose effectiveness in out-of-domain data, we report results under the zero-shot BEIR benchmark in Table 5. We actually observe a small boost in effectiveness on sparse retrieval when using models solely trained on MSMARCO, while on Dense there's a more apparent decrease of performance. The biggest difference for dense models is on the TREC Covid dataset which is far from the MSMARCO collection. Given the nature of the BEIR benchmark (mean over 13 datasets), those differences may not be significative.[2]

[2] We could not find in the literature an easy/practical way to perform statistical significance testing over BEIR.

Table 5. Experiments on zero-shot retrieval on BEIR (nDCG@10) with models finetuned on MSMARCO. Models with † were pretrained solely on MSMARCO.

	SPLADE					Dense				
	Distilbert	Bert	M 6L†	M+F 6L†	M+F 12L†	Distilbert	Bert	M 6L†	M+F 6L†	M+F 12L†
arguana	45.30	44.30	**46.90**	42.90	45.40	34.00	37.30	40.20	37.40	40.70
climate-fever	14.90	15.30	14.50	15.70	15.40	16.50	15.90	15.40	15.30	**17.50**
dbpedia-entity	**39.10**	38.80	38.30	38.80	37.90	31.50	31.70	31.10	31.70	30.60
fever	**73.40**	71.20	68.60	72.30	70.70	70.80	70.20	59.10	59.90	56.10
fiqa	31.20	29.90	**33.20**	32.30	31.70	25.70	24.10	27.30	27.00	27.60
hotpotqa	66.90	65.50	64.00	67.30	**67.60**	49.70	50.20	47.30	46.40	47.60
nfcorpus	33.40	31.30	35.40	**35.90**	34.70	26.40	25.80	28.80	28.70	28.80
nq	**51.40**	50.60	48.50	49.20	49.40	46.00	47.10	41.90	41.50	42.90
quora	77.20	76.60	80.40	78.10	73.80	78.50	82.20	82.20	**83.30**	80.30
scidocs	14.90	14.90	14.40	**15.10**	14.50	11.40	11.80	10.70	11.20	11.00
scifact	66.00	64.70	**69.20**	69.20	69.00	52.00	54.80	56.70	57.30	56.50
trec-covid	67.60	**69.10**	65.60	64.70	68.00	66.10	65.70	56.00	48.90	49.90
webis-touche2020	27.60	27.00	24.70	**28.50**	26.20	22.20	23.50	23.20	19.70	19.30
mean	**46.84**	46.09	46.43	**46.92**	46.48	40.83	41.56	39.99	39.10	39.14

4.3 RQ3: Can we take advantage of that pretraining from scratch in collections of specialized domains/languages

Domain Specific IR on TripClick The TripClick collection [44] contains approximately 1.5 millions MEDLINE documents (title and abstract), and 692,000 queries. The test set is divided into three categories of queries: Head, Torso and Tail (decreasing frequency), which contain 1,175 queries each. For the Head queries a DCTR click model [9] was employed to created relevance signals, the other two sets use raw clicks. [21] showed that the original triplets were too noisy, and released a new training set which we use in this experiment (10 millions triplets).

As pretrained models, we use the off-the-shelf BERT, DistilBERT models, and similarly to [21] SciBERT [3] and PubMedBERT [17] which are both using a similar architecture to Bert (12 layers) and were pretrained using scientific documents (from where they extract their vocabulary). We consider them as off-the-shelf domain-specific models. While for finetuning, we use a batch size of 200 queries and only one negative per query, taking advantage solely of in-batch negatives for 90,000 iterations, which is equivalent to 1.8 epochs.

As Table 6 shows, models pretrained from scratch compete well against generic off-the-shelf models as well as against specialized ones. For this collection, the dense models perform better than the sparse ones. The conclusions depend on the model type: sparse or dense. For the sparse models, we see that at least one model based on pre-training from scratch outperforms off-the-shelf models such as BERT and DistilBERT, as well as both off-the-shelf domain-specific models (Scibert, PubMedBert). Regarding the dense architecture, pretrained models from scratch are on par with off-the-shelf domain-specific models which required much more training data (1.4B tokens for Scibert against 300M for TripClick). Pre-training from scratch allows for selecting the most suitable language model according to the fine-tuning architecture (sparse or dense). In our case a 6 layer language model is far better for SPLADE while a 12 layer better fits a dense

Table 6. Experiment on Tripclick with sparse (SPLADE) and dense models (nDCG@10).† indicates a method pretrained solely on Tripclik.

#	Pretrained Model	SPLADE				Dense			
		Head_{dctr}	Head	Torso	Tail	Head_{dctr}	Head	Torso	Tail
a	Distilbert	22.1^{b}	30.3^{bd}	23.9^{b}	24.8^{bd}	24.9	34.8	29.2	27.5
b	BERT	10.6	14.4	9.6	7.8	25.3	35.3	29.4	28.8^{a}
c	PubMedBert	22.5^{bd}	31.0^{bd}	24.5^{b}	24.3^{bd}	27.1^{abef}	37.6^{abef}	29.9^{e}	$\mathbf{30.8}^{a}$
d	Scibert	21.1^{b}	28.4^{b}	23.0^{b}	22.2^{b}	27.8^{abef}	38.1^{abef}	29.2	30.3^{a}
e	MLM+F 6L†	$\mathbf{26.9}^{abcdfgh}$	$\mathbf{36.7}^{abcdfgh}$	$\mathbf{27.7}^{abcdfgh}$	$\mathbf{27.2}^{bcdfh}$	25.7	35.7	28.3	29.7
f	MLM 6L†	23.1^{bd}	31.9^{bd}	24.7^{b}	23.4^{b}	25.8	36.0	28.9	29.3
g	MLM+F 12L†	23.7^{abd}	32.5^{abd}	26.2^{abd}	26.6^{bdf}	26.7^{ab}	37.5^{abef}	29.9^{e}	30.1^{a}
h	MLM 12L†	24.0^{abcd}	32.6^{abcd}	24.8^{bd}	24.6^{bdf}	$\mathbf{28.0}^{abefg}$	$\mathbf{38.6}^{abef}$	$\mathbf{30.2}^{e}$	30.4^{a}
	Scibert[21]	-	-	-	-	24.3	-	-	-
	PubMebBert [21]	-	-	-	-	23.5	-	-	-
	BM25 [21]	-	-	-	-	14.0	-	-	-

model. We added at the bottom of Table 6 the results from [21], which show that our implementation choices are very competitive[3] for the sparse as well as for the dense models.

Mr. TyDi is a multilingual dataset for monolingual retrieval composed of 11 typologically diverse languages [54]. For this study we focus on the three non-English languages with the most training data on Mr. TyDi: i) Arabic; ii) Russian; iii) Japanese. Note that, there is a large amount of data available for these languages (even outside of Mr. TyDi), but PLMs are not as well studied as in English. The main consensus seems to be that in this case one should focus on multi-lingual data, for which most is based on english as an anchor, as it is the case for many previous works [23,38,37,54,8]. We challenge this notion, by: a) using monolingual models; b) pretraining solely on Mr. TyDi.

We follow the fine-tuning protocol of MContriever [23], where they first fine-tune the model for retrieval on MMarco [5], a translated version of MSMARCO in multiple languages, for which we only use the target language for a given model (Arabic, Russian or Japanese) and then finally fine-tune on the Mr. TyDi collection. In our case we perform a three-step training, first step on MMarco using negatives sampled with the pretrained model (or MContriever for the baseline). We then perform two steps of training on Mr. TyDi, first using negatives extracted first from the MMarco finetuned model and second from the first stage of Mr.TyDi finetuning. Batch composition follows the previous MSMARCO fine-tuning, with a batch size of 3 queries 32 negatives (the actual batch sizes per GPU is thus 102). Finetuning stopped after two epochs.

Results are available in Table 7. Compared to the previous state of the art [23][4], which is a dense retriever pretrained in a specific fashion on a much larger collection[5], we show statistically significant improvements on all languages

[3] We were not able to find the parameters used in the experiments.

[4] Note that MContriever TyDi (first row) is not available, statistical tests cannot be performed. We do our best to evaluate fairly under our training setting (second row)

[5] We suspect they use more compute, but could not find accurate compute information.

while using solely the Mr. TyDi and MMarco collections on the target language. However, it is important to note that differently from [23] we did not actually test yet on all languages and thus can only evaluate the pretraining effect on these three languages, which are the three largest from Mr. TyDi.

Table 7. Comparison, on the Mr.TyDi dataset, of models trained from scratch against models pretrained in a large external collection with Contriever.† means a method pretrained solely on MrTiDy.

#	Pretrained Model	Arabic		Russian		Japanese	
		MRR@100	**R@100**	**MRR@100**	**R@100**	**MRR@100**	**R@100**
	MContriever [23]	72.4	94.0	59.7	92.4	54.9	88.8
a	MContriever (reproduced)	72.7	93.6	59.9^{b}	91.6	49.9^{b}	85.2^{b}
b	MLM+FLOPS 6L DPR†	73.4	**94.9**	56.2	91.6	32.9	62.0
c	MLM+FLOPS 6L SPLADE†	$\mathbf{75.7}^{ab}$	94.2	$\mathbf{65.0}^{ab}$	$\mathbf{93.4}^{ab}$	$\mathbf{56.3}^{ab}$	$\mathbf{88.9}^{ab}$

4.4 RQ4: Impact of Architectures

Finally, one advantage of pretraining models from scratch is the fact that we can more easily experiment with different architectures. Indeed, considering that most IR works use a variant of BERT (either RoBERTA, DistilBERT or BERT) it raises a question whether variants of transformer architectures, benchmarked in NLP, could actually improve IR. To address this question, we use the sparse retrieval setting from RQ1, but this time also consider using the DeBERTa architecture [19] which beats BERT on many NLP tasks. Results are presented in Tables 8 and 9. Much to our dismay, we did not actually see major improvements using these architectural changes, we thus leave as future work how to better include these changes within first stage rankers.

5 Conclusion

Foundation models come with the promise to be highly general and modular. It is believed that they contain a wide "knowledge" due to their pretraining on a large collection, which is then believed to be the source of their improved performance. We have examined how this pretraining collection influence the performance of IR models. Our research question was to assess how much of this implicit knowledge, beneficial to the final performance, comes from pretraining on a large external collection. This is why we have experimented on a variety of collections, domains and languages to study how pretraining from scratch actually performed compared to their *de facto* approach of simple finetuning.

While we were expecting the standard pretrained models to work better, we surprisingly revealed that pretraining from scratch works better for first-stage retrieval on MSMARCO, TripClick and several non-English languages on the Mr.

Table 8. Comparison on MSMARCO of first stage sparse neural (SPLADE) models using different architectures. All methods are pretrained solely on MSMARCO.

#	Pretrained Model	R-FLOPS	MSMARCO dev		TREC DL 19		TREC DL 20	
			MRR@10	R@1k	nDCG@10	R@1k	nDCG@10	R@1k
a	BERT MLM+FLOPS 6L	0.72	**0.382**^{d}	0.979	0.698	0.836	0.701	**0.872**
b	BERT MLM+FLOPS 12L	0.97	0.379	**0.980**	0.709	0.835	0.709	0.865
c	DeBERTa MLM+FLOPS 6L	0.81	0.376	0.979	0.700	**0.845**	0.701	0.863
d	DeBERTa MLM+FLOPS 12L	0.79	0.373	0.978	**0.716**	0.839	**0.735**	0.871

Table 9. Comparison on TripClick of first stage sparse (SPLADE) and dense models using different architectures. All methods are pretrained solely on TripClick.

#	Pretrained Model	SPLADE				Dense			
		Head$_{dctr}$	Head	Torso	Tail	Head$_{dctr}$	Head	Torso	Tail
a	BERT MLM+FLOPS 6L	26.9^{bcdefh}	36.7^{bcdfh}	27.7^{bdfh}	27.2^{bdfh}	25.7^{e}	35.7^{e}	28.3	29.7^{e}
b	BERT MLM 6L	23.1	31.9	24.7	23.4	25.8^{e}	36.0^{e}	28.9	29.3
c	BERT MLM+FLOPS 12L	23.7^{f}	32.5^{f}	26.2^{fh}	26.6^{bfh}	26.7^{ef}	37.5^{abef}	29.9^{aef}	30.1^{e}
d	BERT MLM 12L	24.0^{fh}	32.6^{f}	24.8	24.6	**28.0**^{abcefgh}	**38.6**^{abefgh}	**30.2**^{aeh}	**30.4**^{e}
e	DeBERTa MLM FLOPS 6L	26.1^{bcdfh}	35.9^{bcdfh}	29.0^{bcdfh}	28.4^{bcdfh}	24.3	34.0	27.8	27.5
f	DeBERTa MLM 6L	22.2	30.7	23.3	23.3	24.9	34.7	28.7	28.9
g	DeBERTa MLM FLOPS 12L	**27.0**^{bcdfh}	**37.5**^{bcdefh}	**29.4**^{abcdfh}	**28.7**^{abcdfh}	26.6^{ef}	36.7^{ef}	29.2	29.5
h	DeBERTa MLM 12L	22.6	31.3	23.8	23.3	26.4^{ef}	36.4^{ef}	28.6	29.0

TyDi benchmark. In particular, the FLOPS regularization played a critical role in those results, suggesting that regularization or better pretraining techniques could further improve the results. Furthermore, pretrained models from scratch also behave well in the zero shot scenario for sparse models such as SPLADE. Nevertheless, pretraining from a large collection has a slight advantage when training rerankers.

Overall, these results, specific to IR, challenge the foundation model hypothesis for small models, ie that a more general model encapsulating the world knowledge would be better than a smaller one in a specific domain application. Furthermore, our study makes a contribution to the debate between general purpose and specific purpose models. In a way, our experiments showed that less is more. In addition, pretrained language models come also with many challenges such as the societal bias in the data they have been trained on. We hope that our study could convince practitioners, both from industry and academia, to reconsider specific purpose models by pretraining from scratch. Last but not least, doing so enable to better control efficiency, data bias and replicability, which are key research questions for the IR community.

References

1. Aroca-Ouellette, S., Rudzicz, F.: On Losses for Modern Language Models. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 4970–4981. Association for Computational Linguistics, Online (Nov 2020). https://doi.org/10.18653/v1/2020.emnlp-main.403, `https://aclanthology.org/2020.emnlp-main.403`

2. Bai, B., Weston, J., Grangier, D., Collobert, R., Sadamasa, K., Qi, Y., Chapelle, O., Weinberger, K.Q.: Supervised semantic indexing. In: Proceedings of the 18th ACM International Conference on Information and Knowledge Management. pp. 187–196. ACM (2009). https://doi.org/10.1145/1645953.1645979
3. Beltagy, I., Lo, K., Cohan, A.: Scibert: A pretrained language model for scientific text (2019). https://doi.org/10.48550/ARXIV.1903.10676, https://arxiv.org/abs/1903.10676
4. Bommasani, R., Hudson, D.A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M.S., Bohg, J., Bosselut, A., Brunskill, E., Brynjolfsson, E., Buch, S., Card, D., Castellon, R., Chatterji, N., Chen, A., Creel, K., Davis, J.Q., Demszky, D., Donahue, C., Doumbouya, M., Durmus, E., Ermon, S., Etchemendy, J., Ethayarajh, K., Fei-Fei, L., Finn, C., Gale, T., Gillespie, L., Goel, K., Goodman, N., Grossman, S., Guha, N., Hashimoto, T., Henderson, P., Hewitt, J., Ho, D.E., Hong, J., Hsu, K., Huang, J., Icard, T., Jain, S., Jurafsky, D., Kalluri, P., Karamcheti, S., Keeling, G., Khani, F., Khattab, O., Koh, P.W., Krass, M., Krishna, R., Kuditipudi, R., Kumar, A., Ladhak, F., Lee, M., Lee, T., Leskovec, J., Levent, I., Li, X.L., Li, X., Ma, T., Malik, A., Manning, C.D., Mirchandani, S., Mitchell, E., Munyikwa, Z., Nair, S., Narayan, A., Narayanan, D., Newman, B., Nie, A., Niebles, J.C., Nilforoshan, H., Nyarko, J., Ogut, G., Orr, L., Papadimitriou, I., Park, J.S., Piech, C., Portelance, E., Potts, C., Raghunathan, A., Reich, R., Ren, H., Rong, F., Roohani, Y., Ruiz, C., Ryan, J., Ré, C., Sadigh, D., Sagawa, S., Santhanam, K., Shih, A., Srinivasan, K., Tamkin, A., Taori, R., Thomas, A.W., Tramèr, F., Wang, R.E., Wang, W., Wu, B., Wu, J., Wu, Y., Xie, S.M., Yasunaga, M., You, J., Zaharia, M., Zhang, M., Zhang, T., Zhang, X., Zhang, Y., Zheng, L., Zhou, K., Liang, P.: On the opportunities and risks of foundation models (2021). https://doi.org/10.48550/ARXIV.2108.07258, https://arxiv.org/abs/2108.07258
5. Bonifacio, L.H., Campiotti, I., Jeronymo, V., Lotufo, R., Nogueira, R.: mmarco: A multilingual version of the ms marco passage ranking dataset. arXiv preprint arXiv:2108.13897 (2021)
6. Chang, W.C., Yu, F.X., Chang, Y.W., Yang, Y., Kumar, S.: Pre-training tasks for embedding-based large-scale retrieval. In: International Conference on Learning Representations (2020), https://openreview.net/forum?id=rkg-mA4FDr
7. Clinchant, S., Jung, K.W., Nikoulina, V.: On the use of BERT for neural machine translation. In: Proceedings of the 3rd Workshop on Neural Generation and Translation. pp. 108–117. Association for Computational Linguistics, Hong Kong (Nov 2019). https://doi.org/10.18653/v1/D19-5611, https://aclanthology.org/D19-5611
8. Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., Stoyanov, V.: Unsupervised cross-lingual representation learning at scale. arXiv preprint arXiv:1911.02116 (2019)
9. Craswell, N., Zoeter, O., Taylor, M., Ramsey, B.: An experimental comparison of click position-bias models. In: Proceedings of the 2008 international conference on web search and data mining. pp. 87–94 (2008)
10. Dehghani, M., Zamani, H., Severyn, A., Kamps, J., Croft, W.B.: Neural ranking models with weak supervision. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 65–74. SIGIR '17, Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3077136.3080832, https://doi.org/10.1145/3077136.3080832

11. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. CoRR **abs/1810.04805** (2018), http://arxiv.org/abs/1810.04805
12. El-Nouby, A., Izacard, G., Touvron, H., Laptev, I., Jegou, H., Grave, E.: Are large-scale datasets necessary for self-supervised pre-training? arXiv preprint arXiv:2112.10740 (2021)
13. Formal, T., Lassance, C., Piwowarski, B., Clinchant, S.: From distillation to hard negative sampling: Making sparse neural ir models more effective. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 2353–2359. SIGIR '22, Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3477495.3531857, https://doi.org/10.1145/3477495.3531857
14. Gao, L., Callan, J.: Condenser: a pre-training architecture for dense retrieval. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. pp. 981–993. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic (Nov 2021). https://doi.org/10.18653/v1/2021.emnlp-main.75, https://aclanthology.org/2021.emnlp-main.75
15. Gao, L., Callan, J.: Unsupervised corpus aware language model pre-training for dense passage retrieval. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 2843–2853. Association for Computational Linguistics, Dublin, Ireland (May 2022). https://doi.org/10.18653/v1/2022.acl-long.203, https://aclanthology.org/2022.acl-long.203
16. Gao, L., Dai, Z., Callan, J.: 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. pp. 3030–3042. Association for Computational Linguistics, Online (Jun 2021). https://doi.org/10.18653/v1/2021.naacl-main.241, https://aclanthology.org/2021.naacl-main.241
17. Gu, Y., Tinn, R., Cheng, H., Lucas, M., Usuyama, N., Liu, X., Naumann, T., Gao, J., Poon, H.: Domain-specific language model pretraining for biomedical natural language processing. ACM Transactions on Computing for Healthcare **3**(1), 1–23 (jan 2022). https://doi.org/10.1145/3458754, https://doi.org/10.1145%2F3458754
18. Guo, Y., Ma, Z., Mao, J., Qian, H., Zhang, X., Jiang, H., Cao, Z., Dou, Z.: Webformer: Pre-training with web pages for information retrieval. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 1502–1512. SIGIR '22, Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3477495.3532086, https://doi.org/10.1145/3477495.3532086
19. He, P., Liu, X., Gao, J., Chen, W.: Deberta: Decoding-enhanced bert with disentangled attention. In: International Conference on Learning Representations (2021), https://openreview.net/forum?id=XPZIaotutsD
20. Hofstätter, S., Althammer, S., Schröder, M., Sertkan, M., Hanbury, A.: Improving efficient neural ranking models with cross-architecture knowledge distillation (2020)
21. Hofstätter, S., Althammer, S., Sertkan, M., Hanbury, A.: Establishing strong baselines for tripclick health retrieval (2022)

22. Hofstätter, S., Lin, S.C., Yang, J.H., Lin, J., Hanbury, A.: Efficiently Teaching an Effective Dense Retriever with Balanced Topic Aware Sampling. In: Proc. of SIGIR (2021)
23. Izacard, G., Caron, M., Hosseini, L., Riedel, S., Bojanowski, P., Joulin, A., Grave, E.: Towards unsupervised dense information retrieval with contrastive learning (2021)
24. Kaplan, J., McCandlish, S., Henighan, T.J., Brown, T.B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., Amodei, D.: Scaling laws for neural language models. ArXiv **abs/2001.08361** (2020)
25. Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., Yih, W.t.: Dense passage retrieval for open-domain question answering. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 6769–6781. Association for Computational Linguistics, Online (Nov 2020). https://doi.org/10.18653/v1/2020.emnlp-main.550, https://www.aclweb.org/anthology/2020.emnlp-main.550
26. Khattab, O., Zaharia, M.: 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. p. 39–48. SIGIR '20, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3397271.3401075, https://doi.org/10.1145/3397271.3401075
27. Kim, T., Yoo, K.M., Lee, S.g.: Self-guided contrastive learning for BERT sentence representations. 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). pp. 2528–2540. Association for Computational Linguistics, Online (Aug 2021). https://doi.org/10.18653/v1/2021.acl-long.197, https://aclanthology.org/2021.acl-long.197
28. Lassance, C., Clinchant, S.: An efficiency study for splade models. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 2220–2226. SIGIR '22, Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3477495.3531833, https://doi.org/10.1145/3477495.3531833
29. Lin, J., Ma, X.: A few brief notes on deepimpact, coil, and a conceptual framework for information retrieval techniques. CoRR **abs/2106.14807** (2021), https://arxiv.org/abs/2106.14807
30. Lin, J., Nogueira, R., Yates, A.: 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
31. Lin, S.C., Yang, J.H., Lin, J.: 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). pp. 163–173. Association for Computational Linguistics, Online (Aug 2021). https://doi.org/10.18653/v1/2021.repl4nlp-1.17, https://aclanthology.org/2021.repl4nlp-1.17
32. Liu, Z., Shao, Y.: Retromae: Pre-training retrieval-oriented transformers via masked auto-encoder (2022). https://doi.org/10.48550/ARXIV.2205.12035, https://arxiv.org/abs/2205.12035
33. Ma, X., Guo, J., Zhang, R., Fan, Y., Ji, X., Cheng, X.: B-prop: Bootstrapped pre-training with representative words prediction for ad-hoc retrieval. Proceedings of

the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (2021)
34. Ma, X., Guo, J., Zhang, R., Fan, Y., Ji, X., Cheng, X.: Prop: Pre-training with representative words prediction for ad-hoc retrieval. Proceedings of the 14th ACM International Conference on Web Search and Data Mining (2021)
35. Ma, Z., Dou, Z., Xu, W., Zhang, X., Jiang, H., Cao, Z., rong Wen, J.: Pre-training for ad-hoc retrieval: Hyperlink is also you need. Proceedings of the 30th ACM International Conference on Information & Knowledge Management (2021)
36. Muennighoff, N.: Sgpt: Gpt sentence embeddings for semantic search. arXiv preprint arXiv:2202.08904 (2022)
37. Nair, S., Yang, E., Lawrie, D., Duh, K., McNamee, P., Murray, K., Mayfield, J., Oard, D.W.: Transfer learning approaches for building cross-language dense retrieval models. In: European Conference on Information Retrieval. pp. 382–396. Springer (2022)
38. Nair, S., Yang, E., Lawrie, D., Mayfield, J., Oard, D.W.: Learning a sparse representation model for neural clir. Design of Experimental Search & Information REtrieval Systems (DESIRES) (2022)
39. Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., Deng, L.: Ms marco: A human generated machine reading comprehension dataset. In: CoCo@ NIPs (2016)
40. Nogueira, R., Cho, K.: Passage re-ranking with bert (2019)
41. Paria, B., Yeh, C.K., Yen, I.E.H., Xu, N., Ravikumar, P., Póczos, B.: Minimizing flops to learn efficient sparse representations (2020)
42. Qu, Y., Ding, Y., Liu, J., Liu, K., Ren, R., Zhao, W.X., Dong, D., Wu, H., Wang, H.: Rocketqa: An optimized training approach to dense passage retrieval for open-domain question answering. In: In Proceedings of NAACL (2021)
43. Reimers, N., Gurevych, I.: Sentence-bert: Sentence embeddings using siamese bert-networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics (11 2019), http://arxiv.org/abs/1908.10084
44. Rekabsaz, N., Lesota, O., Schedl, M., Brassey, J., Eickhoff, C.: Tripclick: The log files of a large health web search engine. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 2507–2513 (2021). https://doi.org/10.1145/3404835.3463242
45. Ren, R., Qu, Y., Liu, J., Zhao, W.X., She, Q., Wu, H., Wang, H., Wen, J.R.: RocketQAv2: A joint training method for dense passage retrieval and passage re-ranking. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. pp. 2825–2835. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic (Nov 2021). https://doi.org/10.18653/v1/2021.emnlp-main.224, https://aclanthology.org/2021.emnlp-main.224
46. Robertson, S.E., Walker, S., Beaulieu, M., Gatford, M., Payne, A.: Okapi at trec-4. Nist Special Publication Sp pp. 73–96 (1996)
47. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108 (2019)
48. Santhanam, K., Khattab, O., Saad-Falcon, J., Potts, C., Zaharia, M.: Colbertv2: Effective and efficient retrieval via lightweight late interaction (2021)
49. Tay, Y., Dehghani, M., Gupta, J.P., Aribandi, V., Bahri, D., Qin, Z., Metzler, D.: Are pretrained convolutions better than pretrained transformers? In: Proceedings of the 59th Annual Meeting of the Association for Computational Lin-

guistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). pp. 4349–4359. Association for Computational Linguistics, Online (Aug 2021). https://doi.org/10.18653/v1/2021.acl-long.335, https://aclanthology.org/2021.acl-long.335

50. Tay, Y., Dehghani, M., Rao, J., Fedus, W., Abnar, S., Chung, H.W., Narang, S., Yogatama, D., Vaswani, A., Metzler, D.: Scale efficiently: Insights from pre-training and fine-tuning transformers. ArXiv **abs/2109.10686** (2022)
51. Thakur, N., Reimers, N., Rücklé, A., Srivastava, A., Gurevych, I.: BEIR: A heterogenous benchmark for zero-shot evaluation of information retrieval models. CoRR **abs/2104.08663** (2021), https://arxiv.org/abs/2104.08663
52. Wu, Y., Schuster, M., Chen, Z., Le, Q.V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Łukasz Kaiser, Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., Dean, J.: Google's neural machine translation system: Bridging the gap between human and machine translation (2016)
53. Xiong, L., Xiong, C., Li, Y., Tang, K.F., Liu, J., Bennett, P.N., Ahmed, J., Overwikj, A.: Approximate nearest neighbor negative contrastive learning for dense text retrieval. In: International Conference on Learning Representations (2021), https://openreview.net/forum?id=zeFrfgyZln
54. Zhang, X., Ma, X., Shi, P., Lin, J.: Mr. tydi: A multi-lingual benchmark for dense retrieval. In: Proceedings of the 1st Workshop on Multilingual Representation Learning. pp. 127–137 (2021)