From Distillation to Hard Negative Sampling: Making Sparse Neural IR Models More Effective

Thibault Formal
thibault.formal@naverlabs.com
Naver Labs Europe
Meylan, France
Sorbonne Université, ISIR
Paris, France

Carlos Lassance
carlos.lassance@naverlabs.com
Naver Labs Europe
Meylan, France

Benjamin Piwowarski
benjamin@piwowarski.fr
Sorbonne Université, ISIR, CNRS
Paris, France

Stéphane Clinchant
stephane.clinchant@naverlabs.com
Naver Labs Europe
Meylan, France

ABSTRACT

Neural retrievers based on dense representations combined with Approximate Nearest Neighbors search have recently received a lot of attention, owing their success to distillation and/or better sampling of examples for training – while still relying on the same backbone architecture. In the meantime, sparse representation learning fueled by traditional inverted indexing techniques has seen a growing interest, inheriting from desirable IR priors such as explicit lexical matching. While some architectural variants have been proposed, a lesser effort has been put in the training of such models. In this work, we build on SPLADE – a sparse expansion-based retriever – and show to which extent it is able to benefit from the same training improvements as dense models, by studying the effect of distillation, hard-negative mining as well as the Pre-trained Language Model initialization. We furthermore study the link between effectiveness and efficiency, on in-domain and zero-shot settings, leading to state-of-the-art results in both scenarios for sufficiently expressive models.

CCS CONCEPTS

• **Information systems → Retrieval models and ranking**.

KEYWORDS

neural networks, indexing, sparse representations, regularization

ACM Reference Format:
Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2022. 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 (SIGIR '22), July 11–15, 2022, Madrid, Spain.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3477495.3531857

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

ACM ISBN 978-1-4503-8732-3/22/07…$15.00
https://doi.org/10.1145/3477495.3531857

1 INTRODUCTION

Traditional IR systems like BM25 have dominated search engines for decades [1], relying on lexical matching and inverted indices to perform efficient retrieval. Since the release of large Pre-trained Language Models (PLM) like BERT [4], Information Retrieval has witnessed a radical paradigm shift towards contextualized semantic matching, where neural retrievers are able to fight the long-standing vocabulary mismatch problem. In the first-stage ranking scenario, dense representations combined with Approximate Nearest Neighbors (ANN) search have become the standard approach, owing their success to improved training pipelines. While these models have demonstrated strong in-domain performance, their ability to generalize has recently been challenged on the recent zero-shot evaluation BEIR benchmark [26], where their average effectiveness is lower than BM25 on a set of various IR-related tasks.

In the meantime, there has been a growing interest in going back to the "lexical space", by learning sparse representations than can be coupled with inverted indexing techniques. These approaches, which generally learn term weighting and/or expansion, benefit from desirable IR priors such as explicit lexical matching and decades of works on optimizing the efficiency of inverted indices. These have also shown good generalization capabilities – w.r.t. either effectiveness [26] or IR behavior like exact match [5]. While they mostly differ in their architectural design, a lesser effort has been put in the training of such models, making it unclear how they would be able to take advantage of the same improvements as dense architectures. In this work, we build on the SPLADE model [6], and study the effect of adopting the latest advances for training dense retrievers. We provide an extensive experimental study – by training models in various scenarios – and illustrate the interplay between models capacity (as reflected by their sparsity) and performance. We show how improvements are additive, and how we are able to obtain state-of-the-art results for sufficiently expressive models.

2 RELATED WORKS

Replacing term-based approaches for candidate generation in search engine pipelines requires models and techniques that can cope with the high latency constraints of serving thousands of queries per

second. Current approaches based on PLM either rely on dense representations combined with ANN search, or sparse representations that can benefit from the proven efficiency of inverted indices.

Dense Representation Learning. Dense retrieval has become the most common approach to apply PLM retrievers in IR or Question Answering. While dense models are generally similar – e.g. relying on the `[CLS]` token – various training strategies have recently been proposed to improve learned representations, ranging from distillation [9, 10, 15, 24], hard negative mining [20, 23, 30, 31], pre-training [7, 11] or any combination of the latter. These models – including for instance DPR [12], ANCE [30] or TAS-B [10] – have recently been challenged on generalization tasks, as shown in the BEIR benchmark [26]. Among dense approaches, ColBERT [13] in contrast relies on a late-interaction mechanism, allowing to perform fine-grained term-level matching at the expense of higher indexing cost and latency.

Sparse Representation Learning. In the meantime, sparse representations based on PLM have seen a growing interest, as they by design inherit from good properties of lexical models. These approaches generally rely on a term-importance component and/or a term-expansion mechanism. Various designs have been proposed in order to cope with one or both aspects. COIL [8] learns term-level dense representations to perform contextualized lexical match; uniCOIL [14] further simplifies the approach by learning a single weight per term, extending previous methods like DeepCT [3] and DeepImpact [16]. On the other hand, SPLADE [6] directly learns high-dimensional sparse representations that are able to jointly perform expansion and re-weighting through the Masked Language Modeling head of the PLM and the help of sparse regularization.

Motivation. While dense approaches have benefited from various improved training strategies, it is unclear whether such improvements could also be observed for sparse models. We wonder if these improvements are *additive*, in the sense that if a model is better than another in a "normal" training setting, would we still observe the same hierarchy in a distillation scenario? Answering such a question would allow to decouple architectures from training innovations when comparing neural retrievers. The recent ColBERTv2 [24] demonstrates how ColBERT can leverage distillation and in-batch negatives to set-up the new state of the art on MS MARCO. In this paper, we follow a similar line: we build on SPLADE and extensively study the influence of various training strategies on model performance, for in-domain and zero-shot settings.

3 SPLADE AND METHODOLOGY

In this section, we first describe in details SPLADE [6], alongside various modifications that can be made in order to improve the model, from distillation and hard negative mining tricks, to the choice of the PLM.

3.1 SPLADE

Model. SPLADE is an expansion-based sparse retrieval model which predicts term importance in the BERT WordPiece vocabulary, relying on predictions from the Masked Language Modeling layer used at pre-training to implicitly perform term expansion. For a query or document t, let $w_{i,j}$ denote the resulting importance of the j-th vocabulary token, for the input token i. Text representations are obtained by pooling such importance predictors over the input sequence, after a log-saturation effect. In SPLADE [5], `sum` pooling is used, but we found out experimentally that using `max` pooling, inspired by [13], led to major improvements over SPLADE (see Section 4). We thus consider by default the following formulation:

$$w_j = \max_{i \in t} \log\left(1 + \text{ReLU}(w_{ij})\right) \tag{1}$$

and the ranking score $s(q, d)$ is given by dot product between q and d representations.

Training. Given a query q, a positive document d^+, a negative document d^- *mined from BM25*, as well as additional in-batch negatives d_j^- (i.e. documents from other queries in the given batch), the model is trained by jointly optimizing a constrastive `InfoNCE` loss [27] – similarly to several prior works on learning first-stage rankers – and the FLOPS regularization [19] directly applied on representations, to obtain the desired sparsity for indices used at retrieval time:

$$\mathcal{L} = \mathcal{L}_{\texttt{InfoNCE},BM25} + \lambda_q \mathcal{L}^q_{\texttt{FLOPS}} + \lambda_d \mathcal{L}^d_{\texttt{FLOPS}} \tag{2}$$

3.2 Distillation, hard negative mining and PLM initialization

In the following, we detail several training tricks previously introduced for dense models that can seamlessly be applied to SPLADE. All the extensions rely on a simple modification of Eq. 2, either by modifying the ranking loss, the source of hard negatives, or a combination of both. We also discuss the initialization strategy, where models can further be improved by relying on PLM checkpoints that have been pre-trained on retrieval-oriented tasks.

Distillation. Distillation has been shown to greatly improve the effectiveness of various neural ranking models [9, 10, 15, 24]. We rely on the MarginMSE loss [9] – MSE between the positive-negative margins of a cross-encoder teacher and the student, and train our model by optimizing[1]:

$$\mathcal{L} = \mathcal{L}_{\texttt{MarginMSE},BM25} + \lambda_q \mathcal{L}^q_{\texttt{FLOPS}} + \lambda_d \mathcal{L}^d_{\texttt{FLOPS}} \tag{3}$$

This distillation setting will be the default for all the scenarios introduced below.

Mining hard negatives. The standard setting using BM25 negatives is limited, and the benefit of using better negative sampling has been highlighted in several prior works [20, 23, 30, 31]. Note that, by considering MarginMSE distillation as the basis for hard negative mining, we remove the need to resort on denoising, i.e. filtering noisy false negatives [20, 23].

Self-mining. Following ANCE [30], which dynamically samples negatives from the model that is being trained, we propose to follow a simpler two-step strategy that has also been adopted in prior works [15]:

- (`step 1`) We initially train a SPLADE model, as well as a cross-encoder re-ranker, in the previously introduced distillation setting;

[1] Cross-encoder teacher scores provided at https://github.com/sebastian-hofstaetter/neural-ranking-kd

- (step 2) We then generate triplets using SPLADE from step 1, and use the cross-encoder to generate the scores needed for MarginMSE, and go for another round of training.

This simply leads to a distillation strategy where mined pairs are supposed to be of better "quality" compared to BM25:

$$\mathcal{L} = \mathcal{L}_{\texttt{MarginMSE},self} + \lambda_q \mathcal{L}^q_{\texttt{FLOPS}} + \lambda_d \mathcal{L}^d_{\texttt{FLOPS}} \quad (4)$$

Ensemble-mining. While the self-mining scenario provides a better sampling strategy compared to BM25, it is rather limited as one could wonder if using various types of models to mine negatives for step 2 could be beneficial. We rely on the recently released msmarco-hard-negatives dataset[2], available in the Sentence Transformers library [22], containing for each query: (1) the top-50 hard negatives mined from BM25 and a set of 12 various dense retrievers, (2) the scores coming from a cross-encoder for each available (q, d^+, d^-) in order to perform MarginMSE knowledge distillation.

$$\mathcal{L} = \mathcal{L}_{\texttt{MarginMSE},ensemble} + \lambda_q \mathcal{L}^q_{\texttt{FLOPS}} + \lambda_d \mathcal{L}^d_{\texttt{FLOPS}} \quad (5)$$

Pre-training. NLP has recently borrowed ideas from contrastive learning techniques in Computer Vision, with the goal of learning high-quality sentence or document representations *without annotation* [7, 11, 21, 28, 29]. The general idea consists in designing pre-training tasks, that are better suited for subsequently training neural retrievers. Most approaches in IR are very similar to CoCondenser [7], which constrastively learns the embedding space from spans of documents (two spans from the same document are considered as positives, the other documents in the batch as negatives). We thus simply consider to use such pre-trained checkpoint to initialize SPLADE, in the scenarios described above. Note that while CoCondenser is not directly optimized for sparse retrieval, its learned embedding space might contain more informative knowledge for retrieval compared to models pre-trained with Masked Language Modeling.

4 EXPERIMENTS AND EVALUATION

We conduct an extensive experimental study, by training and evaluating several models, for each scenario introduced in Section 3. As we provide improvements over SPLADE [6], we refer to all our strategies under the same alias, coined SPLADE++.

4.1 Training and evaluation

Training. All the models are trained on the MS MARCO passage ranking dataset[3], which contains 8.8M passages and 500k training queries with shallow annotations. We follow the same training and evaluation workflow described in [6]. Especially, we use the same hyperparameters, as well as the validation strategy, but modify the components introduced in Section 3 – accordingly the ranking loss for distillation, training files for hard negatives, and/or the PLM initialization. We thus consider the following scenarios: SPLADE, which corresponds to the original setting [6]; DistilMSE which relies on Eq. 3 for training; SelfDistil where models are trained using Eq. 4; EnsembleDistil which rather makes use of Eq. 5; and finally CoCondenser-SelfDistil and CoCondenser-EnsembleDistil which correspond to the two latter scenarios, where the model has additionally been initialized from a pre-trained CoCondenser checkpoint[4].

For each scenario, we train 5 models, each configuration corresponding to different values of the regularization magnitude (λ_q, λ_d) in e.g. Eq. 2, thus providing various trade-offs between effectiveness and efficiency – the higher the λs, the sparser the representations. Note that, as each procedure either modifies the loss or the input pairs, the range taken by loss values slightly differs. We therefore need to adapt the λs for each scenario; we simply rely on grid-search, and kept five configurations which covered a broad range of effective and efficient models[5][6]. To assess model robustness to random training perturbations, we train and evaluate each configuration *three times* – corresponding to different random *seeds*. We thus provide average measures alongside standard deviations.

Evaluation. We evaluate models on the MS MARCO development set which contains 6980 queries, as well as the TREC DL 2019 set which provides fine-grained annotations from human assessors for a set of 43 queries [2]. We report R@1k for both datasets, as well as MRR@10 and nDCG@10 for MS MARCO dev set and TREC DL 2019 respectively. We additionally assess the zero-shot performance of our models on the BEIR benchmark [26]. For comparison with other approaches, we rely on the subset of 13 datasets that are readily available, thus we do not consider CQADupstack, BioASQ, Signal-1M, TREC-NEWS and Robust04; otherwise, we evaluate our models on the complete benchmark (18 datasets).

As SPLADE models allow for various trade-offs – depending on the regularization strength – we resort to the FLOPS measure used in [6] to compare efficiency between models, which gives an estimation of the average number of floating point operations needed to compute the score between a query and a document, empirically estimated from the collection and a set of 100k development queries after training. Note that more informative metrics (e.g. query latency) could have been used. However, such measures can be hard to properly evaluate, depending on the systems; as we only compare efficiency of SPLADE models, the FLOPS is therefore informative enough.

4.2 Results and discussion

We compare our models to two types of approaches – reporting results from corresponding papers: i) Simple training relying on a single round of training with BM25 negatives; it includes models like DeepImpact [16], ColBERT [13] or SPLADE [6]; and ii) Training ++ with various training strategies to improve performance; it includes models like ANCE [30], TAS-B [10] or ColBERT v2 [24]. We also include results from Contriever [11] in the zero-shot setting (see Table 2).

MS MARCO dev and TREC DL 2019 results are given in Table 1. For each, we report the best model configuration (among the five that are trained), *with a FLOPS value inferior to* 3. It thus does not necessarily correspond to the best performance we can obtain, but

[2] https://huggingface.co/datasets/sentence-transformers/msmarco-hard-negatives
[3] https://github.com/microsoft/MSMARCO-Passage-Ranking
[4] Available via huggingface: https://huggingface.co/Luyu/co-condenser-marco
[5] Code for training and evaluating models is available at: https://github.com/naver/splade
[6] Checkpoints for models can be found on Hugging Face: https://huggingface.co/naver

Table 1: Evaluation on MS MARCO passage retrieval (dev set) and TREC DL 2019. As each scenario includes 5 models with different regularization strength, we only report the best performance with a FLOPS value inferior to 3 (see Figure 1). As each run is trained with 3 different random seeds, we also report average measures.

model	MS MARCO dev		TREC DL 2019	
	MRR@10	R@1k	nDCG@10	R@1k
Simple training				
BM25	18.4	85.3	50.6	74.5
doc2query-T5 [18]	27.7	94.7	64.2	82.7
DeepImpact [16]	32.6	94.8	69.5	-
SPLADE [6]	32.2	95.5	66.5	81.3
COIL-full [8]	35.5	96.3	70.4	-
ColBERT [13]	36.8	96.9	-	-
Distillation, negative mining or pre-training				
ANCE [30]	33.0	95.9	64.8	-
TCT-ColBERT [15]	35.9	97.0	71.9	76.0
TAS-B [10]	34.7	97.8	71.7	84.3
RocketQA-v2 [23]	38.8	98.1	-	-
CoCondenser [7]	38.2	98.4	-	-
AR2 [32]	39.5	98.6	-	-
ColBERTv2 [24]	39.7	98.4	-	-
Our methods: SPLADE++				
SPLADE (simple training)	34.2	96.6	69.9	81.5
DistilMSE	35.8	97.8	72.9	85.9
SelfDistil	36.8	98.0	72.3	86.9
EnsembleDistil	36.9	97.9	72.1	86.5
CoCondenser-SelfDistil†	37.5	98.4	73.0	87.8
CoCondenser-EnsembleDistil‡	38.0	98.2	73.2	87.5

to a more realistic trade-off between effectiveness and efficiency (i.e. model compression which depends on λ). The interplay between the two is given in Fig. 1 and Fig. 2: we respectively report MRR@10 on MS MARCO dev and mean nDCG@10 on BEIR datasets *vs* FLOPS, for the five configurations in each scenario. Please note that for BEIR, averaging metrics over multiple datasets is questionable [25]: we thus provide evaluation on every dataset in Table 4 in the Appendix section.

Overall, we observe that: (1) SPLADE is able to take advantage of various training strategies to increase its effectiveness; (2) Performance boosts are additive; (3) Our scenarios lead to competitive and state-of-the-art results on respectively in-domain and zero-shot evaluation, e.g. our `CoCondenser-EnsembleDistil` reaches 38 MRR@10 on MS MARCO; (4) Model effectiveness is linked to efficiency (the sparser, the less effective, which is also true when evaluating on zero-shot).

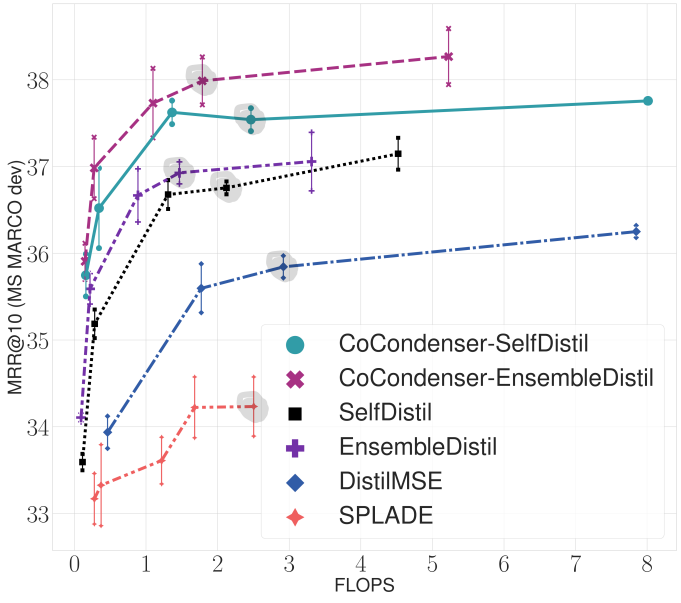
Results analysis. From Table 1, we observe that our SPLADE model with `max` pooling (Eq. 1) already achieves high performance, compared to models in the `simple training` case. The use of distillation (`DistilMSE` scenario) offers the largest boost in effectiveness across all scenarios (+1.6 MRR@10). When combining distillation with hard negative mining, we note that the `SelfDistil` case seems to be the most effective. However, when changing the initialization checkpoint to CoCondenser, we observe the reverse trend, where `CoCondenser-EnsembleDistil` is able to outperform its counterpart. One plausible explanation might be that this checkpoint builds on the `bert-base-uncased` model – contrary to our default `distilbert-base-uncased`, the former containing almost twice as much parameters, allowing to better take advantage of various sources of negative mining. We also note that improvements are less clear when inspecting other metrics like R@1k or nDCG@10 on TREC DL 2019.

Table 2: Mean nDCG@10 on the subset of 13 BEIR datasets. SPLADE++‡,† respectively correspond to our best scenarios CoCondenser-EnsembleDistil‡ and CoCondenser-SelfDistil.†

	BM25	TAS-B	Contriever	ColBERTv2	SPLADE++‡	SPLADE++†
nDCG@10	43.7	43.7	47.5	49.7	50.5	50.7

In Table 2, we report the BEIR results for our two CoCondenser scenarios, reaching state-of-the-art results on zero-shot evaluation, strengthening the observation that sparse retrieval models seem to be better able to generalize [5, 17, 26]. We also report in Table 3 the results of the two previous approaches combined with BM25 (sum); additional gains can be obtained, showing that pure lexical approaches are still somehow complementary to sparse neural models, especially in a zero-shot setting.

Figure 1: MRR@10 (MS MARCO dev) vs FLOPS for all our scenarios. Each point corresponds to the average – with standard deviation – over three runs with different random seeds. Highlighted points correspond to the models for which results are reported in Table 1 (i.e. with FLOPS$\leq$ 3).



	SPLADE++‡ + BM25	SPLADE++† + BM25
nDCG@10	52.1	52.1

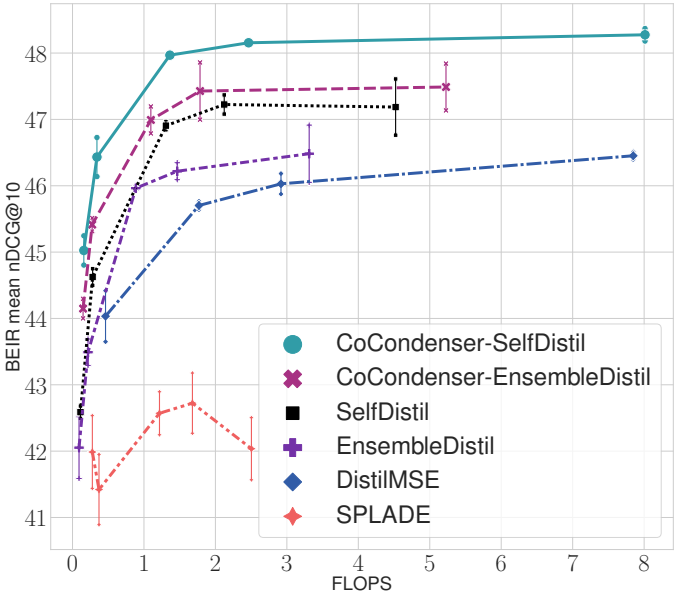
Table 3: Mean nDCG@10 on the subset of 13 BEIR datasets for a simple combination (sum) of SPLADE++ and BM25.

In Fig. 1, we analyse the interplay between effectiveness and efficiency (in terms of FLOPS) on MS MARCO dev set. There is an overall trend that more expressive models tend to be more effective. We also observe additivity in improvements, with the best model on MS MARCO (`CoCondenser-EnsembleDistil`) taking advantage of distillation, ensemble mining and pre-trained checkpoint altogether. We observe the same kind of behavior on the BEIR zero-shot evaluation, where effectiveness comes with a higher complexity (Fig. 2). However, the `SelfDistil` scenarios seem to be better suited for generalization (they both respectively outperform their `EnsembleDistil` counterparts, also see Table 2). One reason for this behavior might be the use of only dense retrievers in the latter case: as such models have been shown to overfit on MS MARCO, the mined negatives might be too specialized on this dataset, thus hurting generalization capabilities of subsequently trained models. Finally, we see that overall, results are rather stable under varying random seeds.

5 CONCLUSION

In this paper, we have built on the SPLADE model, and studied to which extent it is able to take advantage of training improvements like distillation and hard negative mining, as well as better suited PLM initialization: combined altogether, the resulting model reaches state-of-the-art performance on both in-domain and zero-shot evaluation. We also investigated the link between effectiveness and efficiency – induced by the degree of regularization – highlighting that more expressive models are better at generalization.

Figure 2: Mean nDCG@10 over the 18 BEIR datasets vs FLOPS. Each point corresponds to the average – with standard deviation – over three runs with different random seeds.



REFERENCES

[1] Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. 1999. *Modern Information Retrieval.* ACM Press / Addison-Wesley. http://www.ischool.berkeley.edu/~hearst/irbook/glossary.html

[2] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2020. Overview of the trec 2019 deep learning track. *arXiv preprint arXiv:2003.07820* (2020).

[3] Zhuyun Dai and Jamie Callan. 2020. *Context-Aware Term Weighting For First Stage Passage Retrieval.* Association for Computing Machinery, New York, NY, USA, 1533–1536. https://doi.org/10.1145/3397271.3401204

[4] 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

[5] Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021. Match Your Words! A Study of Lexical Matching in Neural Information Retrieval. arXiv:2112.05662 [cs.IR]

[6] Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021. SPLADE: Sparse Lexical and Expansion Model for First Stage Ranking. 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, 2288–2292. https://doi.org/10.1145/3404835.3463098

[7] Luyu Gao and Jamie Callan. 2021. Unsupervised Corpus Aware Language Model Pre-training for Dense Passage Retrieval. *CoRR* abs/2108.05540 (2021). arXiv:2108.05540 https://arxiv.org/abs/2108.05540

[8] Luyu Gao, Zhuyun Dai, and Jamie Callan. 2021. 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.* Association for Computational Linguistics, Online, 3030–3042. https://doi.org/10.18653/v1/2021.naacl-main.241

[9] Sebastian Hofstätter, Sophia Althammer, Michael Schröder, Mete Sertkan, and Allan Hanbury. 2020. Improving Efficient Neural Ranking Models with Cross-Architecture Knowledge Distillation. arXiv:2010.02666 [cs.IR]

[10] 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 *Proc. of SIGIR.*

[11] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Towards Unsupervised Dense Information Retrieval with Contrastive Learning. arXiv:2112.09118

[12] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP).* Association for Computational Linguistics, Online, 6769–6781. https://doi.org/10.18653/v1/2020.emnlp-main.550

[13] 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

[14] Jimmy Lin and Xueguang Ma. 2021. A Few Brief Notes on DeepImpact, COIL, and a Conceptual Framework for Information Retrieval Techniques. *CoRR* abs/2106.14807 (2021). arXiv:2106.14807 https://arxiv.org/abs/2106.14807

[15] Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. 2021. 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).* Association for Computational Linguistics, Online, 163–173. https://doi.org/10.18653/v1/2021.repl4nlp-1.17

[16] Antonio Mallia, Omar Khattab, Torsten Suel, and Nicola Tonellotto. 2021. Learning Passage Impacts for Inverted Indexes. 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, 1723–1727. https://doi.org/10.1145/3404835.3463030

[17] Iurii Mokrii, Leonid Boytsov, and Pavel Braslavski. 2021. *A Systematic Evaluation of Transfer Learning and Pseudo-Labeling with BERT-Based Ranking Models.* Association for Computing Machinery, New York, NY, USA, 2081–2085. https://doi.org/10.1145/3404835.3463093

[18] Rodrigo Nogueira and Jimmy Lin. 2019. From doc2query to docTTTTTquery.

[19] Biswajit Paria, Chih-Kuan Yeh, Ian E. H. Yen, Ning Xu, Pradeep Ravikumar, and Barnabás Póczos. 2020. Minimizing FLOPs to Learn Efficient Sparse Representations. arXiv:2004.05665 [cs.LG]
[20] 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 *In Proceedings of NAACL*.
[21] Ori Ram, Gal Shachaf, Omer Levy, Jonathan Berant, and Amir Globerson. 2021. Learning to Retrieve Passages without Supervision. *CoRR* abs/2112.07708 (2021). arXiv:2112.07708 https://arxiv.org/abs/2112.07708
[22] 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*. Association for Computational Linguistics. https://arxiv.org/abs/1908.10084
[23] Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, QiaoQiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021. 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*. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 2825–2835. https://doi.org/10.18653/v1/2021.emnlp-main.224
[24] 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]
[25] Ian Soboroff. 2018. Meta-Analysis for Retrieval Experiments Involving Multiple Test Collections. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management* (Torino, Italy) *(CIKM '18)*. Association for Computing Machinery, New York, NY, USA, 713–722. https://doi.org/10.1145/3269206.3271719
[26] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A Heterogenous Benchmark for Zero-shot Evaluation of Information Retrieval Models. *CoRR* abs/2104.08663 (2021). arXiv:2104.08663 https://arxiv.org/abs/2104.08663
[27] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation Learning with Contrastive Predictive Coding. *ArXiv* abs/1807.03748 (2018).
[28] Kexin Wang, Nils Reimers, and Iryna Gurevych. 2021. TSDAE: Using Transformer-based Sequential Denoising Auto-Encoder for Unsupervised Sentence Embedding Learning. arXiv:2104.06979 [cs.CL]
[29] Zhuofeng Wu, Sinong Wang, Jiatao Gu, Madian Khabsa, Fei Sun, and Hao Ma. 2020. CLEAR: Contrastive Learning for Sentence Representation. arXiv:2012.15466 [cs.CL]
[30] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwikj. 2021. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. In *International Conference on Learning Representations*. https://openreview.net/forum?id=zeFrfgyZln
[31] Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Min Zhang, and Shaoping Ma. 2020. Learning To Retrieve: How to Train a Dense Retrieval Model Effectively and Efficiently. *CoRR* abs/2010.10469 (2020). arXiv:2010.10469 https://arxiv.org/abs/2010.10469
[32] Hang Zhang, Yeyun Gong, Yelong Shen, Jiancheng Lv, Nan Duan, and Weizhu Chen. 2021. Adversarial Retriever-Ranker for dense text retrieval. https://doi.org/10.48550/ARXIV.2110.03611

A DETAILED BEIR EVALUATION

We provide in Table 4 the complete evaluation on BEIR datasets. Overall, SPLADE variants obtain results that are state-of-the-art.

Table 4: ndcg@10 on BEIR for all the datasets (18). For comparison, we report results directly from corresponding papers, where the evaluation is generally done on the subset of 13 readily available BEIR datasets.

Corpus	BM25	TAS-B	Contriever	ColBERTv2	SPLADE++‡	SPLADE++†
TREC-COVID	65.6	48.1	59.6	**73.8**	72.7	72.5
BioASQ	46.5	38.3	-	-	49.7	**50.8**
NFCorpus	32.5	31.9	32.8	33.8	**34.8**	34.5
NQ	32.9	46.3	49.8	**56.2**	53.7	53.3
HotpotQA	60.3	58.4	63.8	66.7	68.7	**69.3**
FiQA-2018	23.6	30.0	32.9	**35.6**	34.8	34.9
Signal-1M (RT)	**33.0**	28.9	-	-	30.0	30.9
TREC-NEWS	39.8	37.7	-	-	41.5	**41.9**
Robust04	40.8	42.7	-	-	46.7	**48.5**
ArguAna	31.5	42.9	44.6	46.3	**52.5**	51.8
Touché-2020	**36.7**	16.2	23.0	26.3	24.5	24.2
CQADupStack	29.9	31.4	34.5	-	33.4	**35.4**
Quora	78.9	83.5	**86.5**	85.2	83.4	84.9
DBPedia	31.3	38.4	41.3	**44.6**	43.6	43.6
SCIDOCS	15.8	14.9	**16.5**	15.4	15.9	16.1
FEVER	75.3	70.0	75.8	78.5	79.3	**79.6**
Climate-FEVER	21.3	22.8	23.7	17.6	23.0	**23.7**
SciFact	66.5	64.3	67.7	69.3	70.2	**71.0**