SPLADE v2: Sparse Lexical and Expansion Model for Information Retrieval

Thibault Formal
Naver Labs Europe
Meylan, France
Sorbonne Université, LIP6
Paris, France
thibault.formal@naverlabs.com

Benjamin Piwowarski
Sorbonne Université, CNRS, LIP6
Paris, France
benjamin.piwowarski@lip6.fr

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

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

ABSTRACT

In neural Information Retrieval (IR), ongoing research is directed towards improving the first retriever in ranking pipelines. Learning dense embeddings to conduct retrieval using efficient approximate nearest neighbors methods has proven to work well. Meanwhile, there has been a growing interest in learning *sparse* representations for documents and queries, that could inherit from the desirable properties of bag-of-words models such as the exact matching of terms and the efficiency of inverted indexes. Introduced recently, the SPLADE model provides highly sparse representations and competitive results with respect to state-of-the-art dense and sparse approaches. In this paper, we build on SPLADE and propose several significant improvements in terms of effectiveness and/or efficiency. More specifically, we modify the pooling mechanism, benchmark a model solely based on document expansion, and introduce models trained with distillation. We also report results on the BEIR benchmark. Overall, SPLADE is considerably improved with more than 9% gains on NDCG@10 on TREC DL 2019, leading to state-of-the-art results on the BEIR benchmark.

KEYWORDS

neural networks, indexing, sparse representations, regularization

1 INTRODUCTION

The release of large pre-trained language models like BERT [7] has shaken-up Natural Language Processing and Information Retrieval. These models have shown a strong ability to adapt to various tasks by simple fine-tuning. At the beginning of 2019, *Nogueira and Cho* [19] achieved state-of-the-art results – by a large margin – on the MS MARCO passage re-ranking task, paving the way for LM-based neural ranking models. Because of strict efficiency requirements, these models have initially been used as re-rankers in a two-stage ranking pipeline, where first-stage retrieval – or candidate generation – is conducted with bag-of-words models (e.g. BM25) that rely on inverted indexes. While BOW models remain strong baselines [31], they suffer from the long standing vocabulary mismatch problem, where relevant documents might not contain terms that appear in the query. Thus, there have been attempts to substitute standard BOW approaches by learned (neural) rankers. Designing such models poses several challenges regarding efficiency and scalability: therefore there is a need for methods where most of the computation can be done offline and online inference is fast. Dense retrieval with approximate nearest neighbors search has shown impressive results [11, 16, 24, 30], but can still benefit from BOW models (e.g. by combining both types of signals), due to the absence of explicit term matching. Hence, there has recently been a growing interest in learning *sparse representations* for queries and documents [1, 4, 8, 9, 18, 21, 32, 33]. By doing so, models can inherit from the desirable properties of BOW models like exact-match of (possibly latent) terms, efficiency of inverted indexes and interpretability. Additionally, by modeling implicit or explicit (latent, contextualized) *expansion* mechanisms – similarly to standard expansion models in IR – these models can reduce the vocabulary mismatch.

In this paper, we build on the SPLADE model [8] and propose several improvements/modifications that bring gains in terms of effectiveness or efficiency: (1) by simply modifying SPLADE pooling mechanism, we are able to increase effectiveness by a large margin; (2) in the meantime, we propose an extension of the model without query expansion. Such model is inherently more efficient, as everything can be pre-computed and indexed offline, while providing results that are still competitive; (3) finally, we use distillation techniques [12] to boost SPLADE performance, leading to close to state-of-the-art results on the MS MARCO passage ranking task as well as the BEIR zero-shot evaluation benchmark [26].

2 RELATED WORKS

Dense retrieval based on BERT Siamese models [25] has become the standard approach for candidate generation in Question Answering

Conference'17, July 2017, Washington, DC, USA
© 2021 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

and IR [10, 11, 13, 16, 24, 30]. While the backbone of these models remains the same, recent works highlight the critical aspects of the training strategy to obtain state-of-the-art results, ranging from improved negative sampling [11, 16] to distillation [12, 16]. ColBERT [14] pushes things further: the postponed token-level interactions allow to efficiently apply the model for first-stage retrieval, benefiting of the effectiveness of modeling fine-grained interactions, at the cost of storing embeddings for each (sub)term – raising concerns about the actual scalability of the approach for large collections. To the best of our knowledge, very few studies have discussed the impact of using *approximate* nearest neighbors (ANN) search on IR metrics [2, 27]. Due to the moderate size of the MS MARCO collection, results are usually reported with an *exact*, brute-force search, therefore giving no indication on the effective computing cost.

An alternative to dense indexes is term-based ones. Building on standard BOW models, *Zamani et al.* first introduced SNRM [32]: the model embeds documents and queries in a sparse high-dimensional latent space by means of ℓ_1 regularization on representations. However, SNRM effectiveness remains limited and its efficiency has been questioned [22].

Motivated by the success of BERT, there have been attempts to transfer the knowledge from pre-trained LM to sparse approaches. DeepCT [4–6] focused on learning contextualized term weights in the full vocabulary space – akin to BOW term weights. However, as the vocabulary associated with a document remains the same, this type of approach does not solve the vocabulary mismatch, as acknowledged by the use of query expansion for retrieval [4]. A first solution to this problem consists in expanding documents using generative approaches such as doc2query [21] and doc2query-T5 [20] to predict expansion words for documents. The document expansion adds new terms to documents – hence fighting the vocabulary mismatch – as well as repeats existing terms, implicitly performing re-weighting by boosting important terms.

Recently, DeepImpact [18] combined the expansion from doc2query-T5 with the re-weighting from DeepCT to learn term *impacts.* These expansion techniques are however limited by the way they are trained (predicting queries), which is indirect in nature and limit their progress. A second solution to this problem, that has been chosen by recent works [1, 8, 17, 33], is to estimate the importance of each term of the vocabulary *implied by* each term of the document (or query), i.e. to compute an interaction matrix between the document or query tokens and all the tokens from the vocabulary. This is followed by an aggregation mechanism (roughly sum for SparTerm [1] and SPLADE [8], max for EPIC [17] and SPARTA [33]), that allows to compute an importance weight for each term of the vocabulary, for the full document or query.

However, EPIC and SPARTA (document) representations are not sparse enough by construction – unless resorting on top-k pooling – contrary to SparTerm, for which fast retrieval is thus possible. Furthermore, the latter does not include (like SNRM) an *explicit* sparsity regularization, which hinders its performance. SPLADE however relies on such regularization, as well as other key changes, that boost both the efficiency and the effectiveness of this type of approaches, providing a model that both learns expansion and compression in an end-to-end manner. Furthermore, COIL [9] proposed to revisit exact-match mechanisms by learning dense representations *per term* to perform contextualized term matching, at the cost of increased index size.

3 SPARSE LEXICAL REPRESENTATIONS FOR FIRST-STAGE RANKING

In this section, we first describe in details the SPLADE model recently introduced in [8].

3.1 SPLADE

SPLADE predicts term importance – in BERT WordPiece vocabulary ($|V| = 30522$) – based on the logits of the Masked Language Model (MLM) layer. More precisely, let us consider an input query or document sequence (after WordPiece tokenization) $t = (t_1, t_2, ..., t_N)$, and its corresponding BERT embeddings $(h_1, h_2, ..., h_N)$. We consider the importance w_{ij} of the token j (vocabulary) for a token i (of the input sequence):

$$w_{ij} = \text{transform}(h_i)^T E_j + b_j \quad j \in \{1, ..., |V|\} \tag{1}$$

where E_j denotes the BERT input embedding for token j, b_j is a token-level bias, and transform$(.)$ is a linear layer with GeLU activation and LayerNorm. Note that Eq. (1) is equivalent to the MLM prediction, thus it can also be initialized from a pre-trained MLM model. The final representation is then obtained by summing importance predictors over the input sequence tokens, after applying a log-saturation effect [8]:

$$w_j = \sum_{i \in t} \log \left(1 + \text{ReLU}(w_{ij})\right) \tag{2}$$

Ranking loss. Let $s(q, d)$ denote the ranking score obtained via dot product between q and d representations from Eq. (2). Given a query q_i in a batch, a positive document d_i^+, a (hard) negative document d_i^- (e.g. coming from BM25 sampling), and a set of negative documents in the batch (positive documents from other queries) $\{d_{i,j}^-\}_j$, we consider a constrastive loss, which can be interpreted as the maximization of the probability of the document d_i^+ being relevant among the documents d_i^+, d_i^- and $\{d_{i,j}^-\}$:

$$\mathcal{L}_{rank-IBN} = -\log \frac{e^{s(q_i,d_i^+)}}{e^{s(q_i,d_i^+)} + e^{s(q_i,d_i^-)} + \sum_j e^{s(q_i,d_{i,j}^-)}} \tag{3}$$

The *in-batch negatives* (IBN) sampling strategy is widely used for training image retrieval models, and has shown to be effective in learning first-stage rankers [13, 16, 24].

Learning sparse representations. The idea of learning sparse representations for first-stage retrieval dates back to SNRM [32], via ℓ_1 regularization. Later, [22] pointed-out that minimizing the ℓ_1 norm of representations does not result in the most efficient index, as nothing ensures that posting lists are evenly distributed. Note that this is even more true for standard indexes due to the Zipfian nature of the term frequency distribution. To obtain a well-balanced index, *Paria et al.* [22] introduce the `FLOPS` regularizer, a smooth relaxation of the average number of floating-point operations necessary to compute the score between a query and a document, and hence directly related to the retrieval time. It is defined using a_j as a continuous relaxation of the activation (i.e. the term has a non-zero weight) probability p_j for token j, and estimated for documents d in a batch of size N by $\bar{a}_j = \frac{1}{N} \sum_{i=1}^{N} w_j^{(d_i)}$. This gives the following regularization loss

$$\ell_{\texttt{FLOPS}} = \sum_{j \in V} \bar{a}_j^2 = \sum_{j \in V} \left(\frac{1}{N} \sum_{i=1}^{N} w_j^{(d_i)} \right)^2 \quad (4)$$

Overall loss. By jointly optimizing the model in Eq. (2) with ranking and regularization losses, SPLADE combines the best of both worlds for end-to-end training of sparse, expansion-aware representations of documents and queries:

$$\mathcal{L} = \mathcal{L}_{rank-IBN} + \lambda_q \mathcal{L}_{\text{reg}}^q + \lambda_d \mathcal{L}_{\text{reg}}^d \quad (5)$$

where $\mathcal{L}_{\text{reg}}$ is the sparse FLOPS regularization from Eq. 4. We use two distinct regularization weights (λ_d and λ_q) for queries and documents – allowing to put more pressure on the sparsity for queries, which is critical for fast retrieval.

3.2 Pooling strategy

We propose to change the sum in Eq. (2) by a max pooling operation:

$$w_j = \max_{i \in t} \log \left(1 + \text{ReLU}(w_{ij})\right) \quad (6)$$

The model now bears more similarities with SPARTA and EPIC, and to some extent ColBERT. As shown in the experiments section, it considerably improves SPLADE performance. In the following, max pooling is the default configuration for SPLADE, and the corresponding model is referred to as SPLADE-max.

3.3 SPLADE document encoder

In addition to the max pooling operation, we consider a document-only version of SPLADE. In this case, there are no query expansion nor query term weighting, and the ranking score is simply given by:

$$s(q,d) = \sum_{j \in q} w_j^d \quad (7)$$

Such extension offers an interesting efficiency boost: because the ranking score solely depends on the document term weights, everything can be pre-computed offline, and inference cost is consequently reduced, while still offering competitive results as shown in the experiments. We refer to this model as SPLADE-doc.

3.4 Distillation and hard negatives

We also incorporate distillation to our training procedure, following the improvements shown in [12]. The distillation training is done in two steps: (1) we first train both a SPLADE first-stage retriever as well as a cross-encoder reranker [1] using the triplets generated by [12]; (2) in the second step, we generate triplets using SPLADE trained with distillation (thus providing harder negatives than BM25), and use the aforementioned reranker to generate the scores needed for the Margin-MSE loss. We then train a SPLADE model from scratch using these triplets and scores. The result of the second step is what we call DistilSPLADE-max.

[1]Using https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-12-v2 as pre-trained checkpoint.

4 EXPERIMENTAL SETTING AND RESULTS

We trained and evaluated our models on the MS MARCO passage ranking dataset[2] in the full ranking setting. The dataset contains approximately 8.8M passages, and hundreds of thousands training queries with shallow annotation ($\approx$ 1.1 relevant passages per query in average). The development set contains 6980 queries with similar labels, while the TREC DL 2019 evaluation set provides fine-grained annotations from human assessors for a set of 43 queries [3].

Training, indexing and retrieval. We initialized the models with the `DistilBERT-base` checkpoint. Models are trained with the ADAM optimizer, using a learning rate of $2e^{-5}$ with linear scheduling and a warmup of 6000 steps, and a batch size of 124. We keep the best checkpoint using MRR@10 on a validation set of 500 queries, after training for 150k iterations, using an approximate retrieval validation set similar to [11]. For the SPLADE-doc approach, we simply train for 50k steps and select the last checkpoint. We consider a maximum length of 256 for input sequences. In order to mitigate the contribution of the regularizer at the early stages of training, we follow [22] and use a scheduler for λ, quadratically increasing λ at each training iteration, until a given step (50k in our case), from which it remains constant. Typical values for λ fall between $1e^{-1}$ and $1e^{-4}$. For storing the index, we use a custom implementation based on Python arrays, and we rely on Numba [15] to parallelize retrieval. Models[3] are trained using PyTorch [23] and HuggingFace transformers [28], on 4 Tesla $V100$ GPUs with 32GB memory.

Evaluation. We report Recall@1000 for both datasets, as well as the official metrics MRR@10 and NDCG@10 for MS MARCO dev set and TREC DL 2019 respectively. Since we are essentially interested in the first retrieval step, we do not consider re-rankers based on BERT, and we compare our approach to first stage rankers only – results reported on the MS MARCO leaderboard are thus not comparable to the results presented here. We compare to the following sparse approaches (1) BM25, (2) DeepCT [4], (3) doc2query-T5 [20], (4) SparTerm [1], (5) COIL-tok [9] and (6) DeepImpact [18], as well as state-of-the-art dense approaches ANCE [29], TCT-ColBERT [16] and TAS-B [11], reporting results from corresponding papers.

MS MARCO dev and TREC DL 2019 results are given in Table 1: as the performance for SPLADE depends on the regularization strength λ, and as more efficient models are generally less effective, we select in the table the best performing model in our grid of experiments, with reasonable efficiency (in terms of FLOPS). Figure 1 highlights the actual trade-off between effectiveness and efficiency for SPLADE, by showing the performance (MRR@10 on MS MARCO dev set) vs FLOPS, for SPLADE models trained with different regularization strength. The FLOPS metric is 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. Overall, we observe that: (1) *our improved models outperform the other sparse retrieval methods by a large margin*

[2]https://github.com/microsoft/MSMARCO-Passage-Ranking
[3]We made the code public at https://github.com/naver/splade

Table 1: Evaluation on MS MARCO passage retrieval (dev set) and TREC DL 2019.

model	MS MARCO dev		TREC DL 2019	
	MRR@10	R@1000	NDCG@10	R@1000
`Dense retrieval`				
Siamese (ours)	0.312	0.941	0.637	0.711
ANCE [29]	0.330	0.959	0.648	-
TCT-ColBERT [16]	0.359	0.970	0.719	0.760
TAS-B [11]	0.347	0.978	0.717	0.843
RocketQA [24]	0.370	0.979	-	-
`Sparse retrieval`				
BM25	0.184	0.853	0.506	0.745
DeepCT [4]	0.243	0.913	0.551	0.756
doc2query-T5 [20]	0.277	0.947	0.642	0.827
SparTerm [1]	0.279	0.925	-	-
COIL-tok [9]	0.341	0.949	0.660	-
DeepImpact [18]	0.326	0.948	0.695	-
SPLADE [8]	0.322	0.955	0.665	0.813
`Our methods`				
SPLADE-max	0.340	0.965	0.684	0.851
SPLADE-doc	0.322	0.946	0.667	0.747
DistilSPLADE-max	0.368	0.979	0.729	0.865

on the MS MARCO dev set as well as TREC DL 2019 queries; (2) *the results are competitive with state-of-the-art dense retrieval methods.*

BEIR. Finally, we verify the zero-shot performance of SPLADE using a subset of datasets from the BEIR [26] benchmark that encompasses various IR datasets for zero-shot evaluation. We solely use a subset due to the fact that some of the datasets (namely `CQADupstack`, `BioASQ`, `Signal-1M`, `TREC-NEWS`, `Robust04`) are not readily available. Results are displayed in Table 2 (NDCG@10). We compare against the best performing models from the original benchmark paper [26] (ColBERT [14]) and the two best performing from the rolling benchmark[4] (tuned BM25 and TAS-B [11]). We also report the SPLADE evaluation against these baselines.

4.1 Impact of max pooling

First, on MS MARCO and TREC, max pooling brings almost 2 points in MRR@10 and NDCG@10 compared to the SPLADE baseline. It becomes competitive with COIL and DeepImpact. In addition, Figure 1 shows that SPLADE-max is consistently better than SPLADE, in terms of effectiveness-efficiency trade-off. SPLADE-max has also improved performance on the BEIR benchmark (cf Table 2).

4.2 Document expansion

Our document encoder model with max pooling is able to reach the same performance as the previous SPLADE model, outperforming doc2query-T5 on MS MARCO. As this model has no query encoder, it is more efficient in terms of e.g. latency. Figure 2 illustrates how we can balance efficiency (in terms of the average size of document representations) with effectiveness. For relatively sparse representations, we are able to obtain performance on par with approaches like doc2query-T5 (e.g. MRR@10=29.6 for a model with an average of 19 non-zero weights per document). In addition, it is straightforward to train and apply to a new document collection: a single forward is required as opposed to multiple inferences with beam search for doc2query-T5.

4.3 Distillation

By training with distillation, we are able to considerably improve the performance of SPLADE, as seen in Table 1. From Figure 1, we observe that distilled models bring huge improvements for higher

[4]https://docs.google.com/spreadsheets/d/1L8aACyPaXrL8iEelJLGqlMqXKPX2oSP_R10pZoy77Ns

Figure 1: Performance vs FLOPS for SPLADE models trained with different regularization strength λ on MS MARCO.

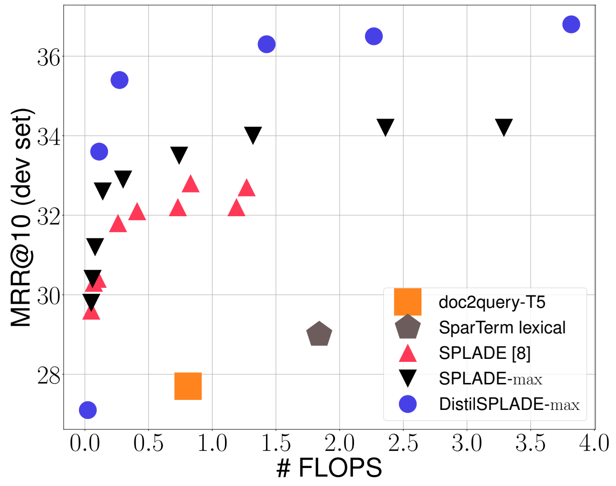

Figure 2: Performance vs average document length (number of non-zero dimensions in document representations) for SPLADE-doc models trained with different regularization strength λ_d on MS MARCO.

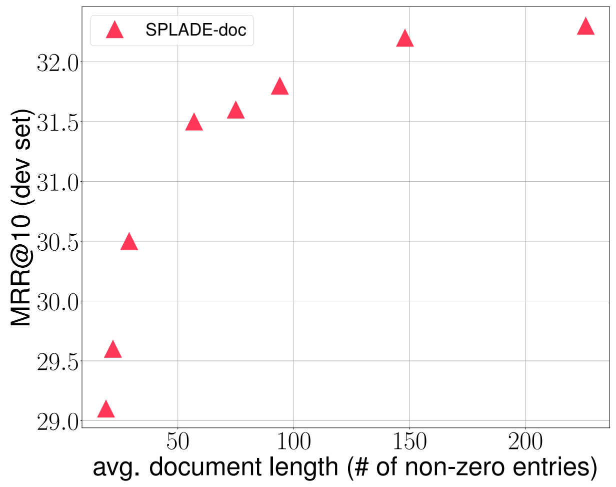

Table 2: NDCG@10 results on BEIR (subset containing all the readily available datasets).

Corpus	Baselines			SPLADE		
	ColBERT	BM25	TAS-B	sum [8]	max	distil
`MS MARCO`	0.425	0.228	0.408	0.387	0.402	**0.433**
`ArguAna`	0.233	0.315	0.427	0.447	0.439	**0.479**
`Climate-FEVER`	0.184	0.213	0.228	0.162	0.199	**0.235**
`DBPedia`	0.392	0.273	0.384	0.343	0.366	**0.435**
`FEVER`	0.771	0.753	0.700	0.728	0.730	**0.786**
`FiQA-2018`	0.317	0.236	0.300	0.258	0.287	**0.336**
`HotpotQA`	0.593	0.603	0.584	0.635	0.636	**0.684**
`NFCorpus`	0.305	0.325	0.319	0.311	0.313	**0.334**
`NQ`	**0.524**	0.329	0.463	0.438	0.469	0.521
`Quora`	**0.854**	0.789	0.835	0.829	0.835	0.838
`SCIDOCS`	0.145	**0.158**	0.149	0.141	0.145	**0.158**
`SciFact`	0.671	0.665	0.643	0.626	0.628	**0.693**
`TREC-COVID`	0.677	0.656	0.481	0.655	0.673	**0.710**
`Touché-2020 (v1)`	0.275	**0.614**	0.173	0.289	0.316	0.364
Avg. all	0.455	0.440	0.435	0.446	0.460	**0.500**
Avg. zero-shot	0.457	0.456	0.437	0.451	0.464	**0.506**
Best on dataset	2	2	0	0	0	**11**

values of FLOPS (0.368 MRR@10 for ≈ 4 FLOPS), but are still very efficient in low regime (0.35 MRR for ≈ 0.3 FLOPS). Furthermore, DistilSPLADE-max is able to outperform all other methods in most datasets of the BEIR benchmark (cf Table 2).

5 CONCLUSION

In this paper, we have built on the SPLADE model by reconsidering its pooling mechanism, and by using standard training techniques such as distillation for neural IR models. Our experiments have shown that the max pooling technique indeed provides a substantial improvement. Secondly, the document encoder is an interesting model for faster retrieval conditions. Finally, the distilled SPLADE model leads to close to state-of-the-art models on MS MARCO and TREC DL 2019, while clearly outperforming recent dense models on zero-shot evaluation.

REFERENCES

[1] Yang Bai, Xiaoguang Li, Gang Wang, Chaoliang Zhang, Lifeng Shang, Jun Xu, Zhaowei Wang, Fangshan Wang, and Qun Liu. 2020. SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval. arXiv:2010.00768 [cs.IR]

[2] Leonid Boytsov. 2018. *Efficient and Accurate Non-Metric k-NN Search with Applications to Text Matching.* Ph.D. Dissertation. Carnegie Mellon University.

[3] 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).

[4] Zhuyun Dai and Jamie Callan. 2019. Context-Aware Sentence/Passage Term Importance Estimation For First Stage Retrieval. arXiv:1910.10687 [cs.IR]

[5] Zhuyun Dai and Jamie Callan. 2020. *Context-Aware Document Term Weighting for Ad-Hoc Search.* Association for Computing Machinery, New York, NY, USA, 1897–1907. https://doi.org/10.1145/3366423.3380258

[6] 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

[7] 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

[8] 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

[9] 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

[10] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-Augmented Language Model Pre-Training. arXiv:2002.08909 [cs.CL]

[11] 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*.

[12] 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]

[13] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. arXiv:2004.04906 [cs.CL]

[14] 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

[15] Siu Kwan Lam, Antoine Pitrou, and Stanley Seibert. 2015. Numba: A llvm-based python jit compiler. In *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*. 1–6.

[16] 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

[17] Sean MacAvaney, Franco Maria Nardini, Raffaele Perego, Nicola Tonellotto, Nazli Goharian, and Ophir Frieder. 2020. Expansion via Prediction of Importance with Contextualization. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Jul 2020). https://doi.org/10.1145/3397271.3401262

[18] 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

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

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

[21] Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document Expansion by Query Prediction. arXiv:1904.08375 [cs.IR]

[22] 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]

[23] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library.. In *NeurIPS*.

[24] 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 *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Online, 5835–5847. https://doi.org/10.18653/v1/2021.naacl-main.466

[25] 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. http://arxiv.org/abs/1908.10084

[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] Zhengkai Tu, Wei Yang, Zihang Fu, Yuqing Xie, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2020. Approximate Nearest Neighbor Search and Lightweight Dense Vector Reranking in Multi-Stage Retrieval Architectures. In *Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval*. 97–100.

[28] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. HuggingFace's Transformers: State-of-the-art Natural Language Processing. arXiv:1910.03771 [cs.CL]

[29] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. arXiv:2007.00808 [cs.IR]

[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] Wei Yang, Kuang Lu, Peilin Yang, and Jimmy Lin. 2019. Critically Examining the "Neural Hype". *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Jul 2019). https://doi.org/10.1145/3331184.3331340

[32] Hamed Zamani, Mostafa Dehghani, W. Bruce Croft, Erik Learned-Miller, and Jaap Kamps. 2018. From Neural Re-Ranking to Neural Ranking: Learning a Sparse Representation for Inverted Indexing. 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, 497–506. https://doi.org/10.1145/3269206.3271800

[33] Tiancheng Zhao, Xiaopeng Lu, and Kyusong Lee. 2020. SPARTA: Efficient Open-Domain Question Answering via Sparse Transformer Matching Retrieval. arXiv:2009.13013 [cs.CL]