CSPLADE: Learned Sparse Retrieval with Causal Language Models

Zhichao Xu, Aosong Feng, Yijun Tian, Haibo Ding, Lin Lee Cheong
Amazon Web Services
xzhichao@amazon.com

Abstract

In recent years, dense retrieval has been the focus of information retrieval (IR) research. While effective, dense retrieval produces uninterpretable dense vectors, and suffers from the drawback of large index size. Learned sparse retrieval (LSR) has emerged as promising alternative, achieving competitive retrieval performance while also being able to leverage the classical inverted index data structure for efficient retrieval. However, limited works have explored scaling LSR beyond BERT scale. In this work, we identify two challenges in training large language models (LLM) for LSR: (1) training instability during the early stage of contrastive training; (2) suboptimal performance due to pre-trained LLM's unidirectional attention. To address these challenges, we propose two corresponding techniques: (1) a lightweight *adaptation* training phase to eliminate training instability; (2) two model variants to enable *bidirectional information*. With these techniques, we are able to train LSR models with 8B scale LLM, and achieve competitive retrieval performance with reduced index size. Furthermore, we are among the first to analyze the performance-efficiency tradeoff of LLM-based LSR model through the lens of model quantization. Our findings provide insights into adapting LLMs for efficient retrieval modeling.

1 Introduction

Recently, the main research focus in information retrieval (IR) has been on dense retrieval and related techniques (Karpukhin et al., 2020; Lin et al., 2022; Zhu et al., 2023a; Xu et al., 2025b, *inter alia*). Dense retrieval encodes queries and documents into high-dimensional sparse vectors. Although effective, these dense vectors are difficult for humans to interpret in terms of their semantic meanings. Moreover, encoding and storing the dense vectors for the whole document collection can be resource-intensive. For example, encoded flat index of MS MARCO passage corpus (Bajaj et al., 2016) with Llama-2-7b dense retriever takes up 135G disk space (Ma et al., 2024), which is over 50 times larger than the 2.6G Lucene index from Lucene's implementation of BM25.

To mitigate these drawbacks of dense retrieval, a different line of works investigates learned sparse retrieval (LSR). Inspired by traditional sparse retrieval models (Sparck Jones, 1972; Robertson et al., 1995), LSR encodes queries and documents into vocabulary-sized vectors with a backbone language model and the language model head, where each dimension of the vector represents the "impact" of the corresponding token (Formal et al., 2021b,a; Mallia et al., 2021). A canonical example of LSR is SPLADE (Formal et al., 2021b,a). It encodes text with BERT (Devlin et al., 2019), then applies pooling and log-saturation (Fang et al., 2004) to ensure the resulting vocabulary-sized vector contains non-negative values in each dimension, making it suitable for use in an inverted index. Combined with established training methodologies in dense retrieval such as contrastive learning (Oord et al., 2018), hard negatives mining (Karpukhin et al., 2020; Xiong et al., 2021) and knowledge distillation (Hofstätter et al., 2020), LSR has demonstrated competitive performance with BERT-style encoder-only masked language models (Kong et al., 2023; Lassance et al., 2024).

Scaling has been a winning recipe for natural language processing (NLP) and IR (Kaplan et al., 2020; Hoffmann et al., 2022; Fang et al., 2024). Recent works (Ma et al., 2024; Lee et al., 2024; Wang et al., 2024a; Xu et al., 2025a; Zhang et al., 2025, *inter alia*) have explored scaling dense retrieval and reranking with pre-trained decoder-only large language models (LLM) such as Llama (Touvron et al., 2023) and Mistral (Jiang et al., 2023), which have demonstrated superior performance compared to BERT family models. However, there has been limited effort in training LSR models beyond BERT scale, i.e., 110M and 330M. In our preliminary experiments, we identified two key challenges in training LSR with decoder-only LLMs: (1) the ReLU activation function used in log-saturation of SPLADE leads to *training instability* problem in early stage of contrastive training, commonly referred to as the dying ReLU problem (Lu et al., 2019); (2) the *unidirectional attention* of decoder-only LLMs leads to suboptimal retrieval performance (Lee et al., 2024; BehnamGhader et al., 2024). To address these two challenges, in this paper we propose two techniques:

- We propose a lightweight *adaptation phase* training, where we adapt the pre-trained language model on unlabeled texts with a combination of causal language modeling loss and log-saturation loss. Experimental results show that as few as 10k adaptation steps can eliminate the training instability problem in subsequent contrastive training.
- We explore two variants to able unidirectional LMs to capture *bidirectional information*: (1) applying the echo embedding idea (Springer et al., 2024), where we repeat the input sequence and only gather the representation from the second occurrence of text sequence; (2) directly disabling the causal language modeling (CLM) mask, and letting the language model adapt to bidirectional information in the contrastive training phase, similar to Lee et al. (2024). Our experiments show that both variants significantly improve upon causal language model with unidirectional information.

We refer to our method as Causal SPLADE (CSPLADE). With the proposed techniques, we are able to train LSR model with up to 8B scale pre-trained LM (Llama-3.1-8B), while achieving competitive performance with only MS MARCO passage retrieval training set (41.3 MRR on MS MARCO passage retrieval, 55.3 NDCG@10 on BEIR) and reduced index size (<8G Lucene index of MS MARCO passage corpus versus 135G flat dense index).

A significant challenge in adopting LLMs for retrieval lies in the scalability and inference latency. We examine several popular quantization methods such as LLM.int8 (Dettmers et al., 2022), torchao (torchao team, 2024), and report the performance-efficiency tradeoff when applying on CSPLADE. We find while calibration-free quantization methods achieve reduced GPU memory usage, they does not necessarily lead to inference speedup in small batch sizes. Our findings underscore the importance of in-depth study of model quantization methods specifically designed and optimized for neural retrieval models.

2 Background and Notations

In this section we introduce the task definition and notations used in this paper, and further provide background for learned sparse retrieval, with a special focus on SPLADE (Formal et al., 2021b,a).

2.1 Task Definition and Notations

Given a query Q, the task is to find a ranked list of k documents, denoted by $\{D_1, D_2, \ldots, D_k\}$, that exhibit high relevance to Q. Retrieval is performed by finding top-k documents from document collection $\mathcal{C}$, where $|\mathcal{C}| \gg k$. We denote the retrieval model parameterized by θ as $f_\theta(\cdot)$. To support efficient retrieval, the document collection is typically pre-encoded offline by the retrieval model, resulting in what is referred to as the *document index*. At retrieval time, the incoming query is first encoded by the retrieval model, after which a similarity search is performed against the pre-built document index.

2.2 Sparse Retrieval

Different from the prevalent dense retrieval method (Karpukhin et al., 2020; Xiong et al., 2021, *inter alia*) that represents a document with a dense vector, the sparse retrieval method represents a document with a vocabulary-sized vector where most of the elements are zeros, hence the term "sparse". This sparse vector representation can be subsequently used in an inverted index for efficient retrieval. Examples of sparse retrieval include classical methods such as the boolean model (Salton, 1984) and probabilistic retrieval models like BM25 (Robertson et al., 1995).

Traditional sparse retrieval methods focus on capturing lexical match signals, which hinders performance is finding semantically relevant documents (Yates et al., 2024). Learned sparse retrieval emerges as a way to leverage pre-trained language models to mitigate this weakness. At a higher level, LSR can be viewed as a way to learn token importance or "impact" scores from data (Dai and Callan, 2019; Bai et al., 2020; Mallia et al., 2021).

2.3 SPLADE

We detail the formulation of SPLADE (Formal et al., 2021b,a), which serves as the basis of the proposed method (Section 3). Denote vocabulary as $\mathcal{V}$, a document as D, tokens as $\{t_1, t_2, \ldots t_{|D|}\}$, where t_i is the i-th token, and its corresponding contextualized representation (e.g., from pre-trained BERT) $\{\mathbf{h}_1, \mathbf{h}_2, \ldots \mathbf{h}_{|D|}\}$. For each $\mathbf{h}_i$, we project the hidden representation to a vocabulary-sized vector $\mathbf{H}_i \in \mathbb{R}^{|\mathcal{V}|}$ with the language modeling head (e.g., masked language modeling head for BERT). The j-th dimension of $\mathbf{H}_i$ represents the importance of token j (in vocabulary $\mathcal{V}$) to token i in the input sequence, which in practice is the logit$_j$ from the LM head output. Given $\mathbf{H}_D = \{\mathbf{H}_1, \mathbf{H}_2, \ldots \mathbf{H}_{|D|}\}$ of tensor shape $(|\mathcal{V}|, |D|)$, SPLADE then applies a max-pooling along the sequence length dimension, i.e., across all tokens, followed by ReLU activation and log rescaling to get the vocabulary-sized representation for the input document d:

$$\mathbf{D} = \log\Big(1 + \text{ReLU}\big(\text{MaxPooling}(\mathbf{H}_D)\big)\Big) \in \mathbb{R}^{|\mathcal{V}|} \tag{1}$$

A similar operation can also be applied to query Q to get query representation $\mathbf{Q} \in \mathbb{R}^{|\mathcal{V}|}$. Denote a similarity function as $s(\cdot)$ (e.g., dot product), we can optimize SPLADE with the standard InfoNCE loss (Oord et al., 2018) for contrastive training. Denote a training pair (Q, D^+), where D^+ is relevant to query Q, and $\{D_N\}$ is a list of documents not relevant to Q, the ranking loss is denoted by:

$$\begin{aligned}\mathcal{L}_{rank}(Q, D^+, \{D_N\}) &= -\log p(D = D^+|Q) \\ &= -\log \frac{e^{s(Q,D^+)}}{e^{s(Q,D^+)} + \sum\limits_{D_i^- \in \{D_N\}} e^{s(Q,D_i^-)}}\end{aligned}$$

In practice, $\{D_N\}$ often includes hard negatives and in-batch negatives (Qu et al., 2021; Ma et al., 2024). [1] Notice that Equation (1) already achieves a certain degree of sparsity by ensuring the non-negativity. In addition, SPLADE also employs `FLOPs` regularization (Paria et al., 2020) to further enhance sparsity in order to learn efficient sparse representation. Denote `FLOPs` regularization loss for Q and D as $\mathcal{L}^Q_{reg}$ and $\mathcal{L}^D_{reg}$, respectively, and λ_Q, λ_D as the corresponding coefficients, SPLADE optimizes the final loss as:

$$\mathcal{L} = \mathcal{L}_{rank}(Q, D^+, \{D_N\}) + \lambda_Q \mathcal{L}^Q_{reg} + \lambda_D \mathcal{L}^D_{reg}$$

[1]Subsequent works (Formal et al., 2021a; Kong et al., 2023) have explored other training strategies such as distillation (Hofstätter et al., 2020). In this study we opted for straightforward contrastive training, as more complex training strategies are orthogonal to the focus of this paper.

In practice, λ_Q and λ_D are tuned as hyperparameters to balance performance and efficiency.

3 Challenges and Proposed Techniques

SPLADE's effectiveness at BERT-scale has been demonstrated by extensive prior studies (Formal et al., 2021a, 2022; Kong et al., 2023; Li et al., 2023, *inter alia*). However, limited studies have explored to train SPLADE beyond BERT-scale, i.e., to extend to pre-trained causal large language models like Llama (Touvron et al., 2023) or Mistral (Jiang et al., 2023) to further improve performance with stronger backbone LMs and extensive pre-training. In our preliminary experiments where we replace BERT (Devlin et al., 2019) in original SPLADE implementation with causal LLMs like OPT-1.3B (Zhang et al., 2022) and Llama-3.2-1B (Grattafiori et al., 2024), we identified key challenges (Section 3.1). We then propose corresponding strategies to enable training SPLADE with causal language models (Section 3.2). We name our method as Causal SPLADE (CSPLADE).

3.1 Challenges

First, we notice the *training instability* problem in early stage of contrastive training. Recall in Equation (1), the ReLU activation function is used to ensure non-negativity of the vocabulary-sized representation $\mathbf{Q}$ and $\mathbf{D}$. As the training starts, ReLU neurons quickly become inactive and only output 0 for any input. This is referred to as the dying ReLU problem in literature (Lu et al., 2019), and is caused because of the initialization of the parameters to be optimized, the backbone language model in our case. To validate this hypothesis, we also attempted to train other encoder-only models including MosaicBERT (Portes et al., 2023) and ModernBERT (Warner et al., 2024). However, training consistently failed for the same reason, despite extensive tuning of the learning rate warmup strategy and other hyperparameters.

Second, we observe that in the original SPLADE implementation, the model first projects the contextualized representation of each token into the vocabulary space to get $|D|$ vectors. It then applies MaxPooling over the sequence length dimension to get a single vector representation. Apparently, this sequence-level MaxPooling operation becomes suboptimal for causal LLMs with *unidirectional attention*, as early tokens in the input sequence cannot attend to later ones, leading to a loss of information in the final vector representation. To summarize, we believe these two challenges hinders the exploration to extend SPLADE to the large-scale pre-trained casual LLMs.

3.2 Proposed Method

We have identified two challenges in Section 3.1, namely, (1) the training instability problem and (2) the unidirectional information problem. To address these challenges, we introduce a two-phase training pipeline to get the final retrieval model: (1) *adaptation phase training* to improve training stability and (2) *enabling bidirectional information* in contrastive learning to capture richer context.

Adaptation Phase Training. As pointed out by Lu et al. (2019), the reason for dying ReLU is due to the ill-initialized model parameters. Therefore we tackle the training instability problem by adapting the pre-trained causal language model for improved initialization to be used in subsequent contrastive training phase. In particular, for an input sequence D, we can compute the output logits $\mathbf{H}_D \in \mathbb{R}^{|\mathcal{V}|\times|D|}$. We then apply the similar transformation in Equation (1), except that we remove the MaxPooling operation:

$$\mathbf{H}_D^* = \log\left(1 + \text{ReLU}(\mathbf{H}_D)\right)$$

Here $\mathbf{H}_D^*$ tensor has the same shape as $\mathbf{H}_D$, but all elements are non-negative. We use this non-negative tensor to compute causal language model loss to maximize the probability of the target sequence (the same input sequence in our case). Denote this causal language modeling loss intended for ReLU adaptation as $\mathcal{L}_{ReLU}$, and the vanilla causal language model loss as $\mathcal{L}_{CLM}$, we optimize a final loss:

$$\mathcal{L}_{Adapt} = \mathcal{L}_{CLM} + \lambda_{ReLU} \cdot \mathcal{L}_{ReLU}$$

where λ_{ReLU} is a trade-off weight balancing two loss terms, which in practice is set to 1. Note here the vanilla causal language modeling loss is computed directly between $\mathbf{H}_D$ and target sequence. Therefore we can compute the two loss terms with one single forward pass. This loss form enables us to adapt the pre-trained causal LLM for the ReLU activation used later in contrastive training. For ease of understanding, we show the pseudo code (PyTorch style) in Appendix A.

This adaptation training strategy bears two strengths: (1) it can be performed on any unlabeled texts; (2) it can be efficiently combined with domain adaptation training (Gururangan et al., 2020) to further improve downstream task performance in the target domain. Empirically, we find as few as 10k steps of adaptation eliminate the training instability problem, suggesting its efficacy.

Enabling Bidirectional Information. Having addressed the training instability problem, we move on to improve learning representations. As we previously discussed (Section 3.1), the unidirectional nature of pre-trained causal language models hinders the capability to effectively learn sequence representations. Different works have attempted to tackle this problem in dense retrieval (Springer et al., 2024; Lee et al., 2024; BehnamGhader et al., 2024, *inter alia*). Motivated by these prior efforts, we study two variants of enabling bidirectional information for pre-trained causal LLMs: (1) *echo embedding* (Springer et al., 2024) and (2) *directly enabling bidirectional attention* (Lee et al., 2024).

Echo embedding does not change the model architecture of pre-trained LLM, but instead changes the input. It repeats the input sequence twice, and only perform the pooling on the second occurrence of the input sequence. This way, the earlier tokens of the input sequence in the second occurrence can still attend to later tokens in the first sequence, enabling bidirectional information. On the other hand, it essentially doubles the input length, making it more computationally extensive in training and also higher latency in inference.

We also experiment with the idea in Lee et al. (2024), where we directly enable the bidirectional attention of pre-trained causal LLM by removing the causal mask, then use the model in subsequent contrastive training. This method merges the phase of adapting for bidirectional information into contrastive training phase, and does not cause additional computational overhead compared to the echo embedding variant. Empirical results show that both variants significantly outperform the causal LLM baseline, with the variant using bidirectional attention yielding slightly superior performance.

We train three variants of SPLADE model with Llama-3 (Grattafiori et al., 2024) as the backbone causal language models, although the techniques discussed can also be applied to other pre-trained causal LLMs. We refer to SPLADE with unidirectional information as CSPLADE (only with adaptation training), and the two bidirectioanl variants as

CSPLADE-Echo and CSPLADE-Bi, respectively.

4 Experiment Setup

In this section we discuss datasets, baselines, implementation and other experimental details.

4.1 Datasets

We focus on the task of passage retrieval. We train the retrieval models on the training split of MS MARCO passage retrieval dataset (Bajaj et al., 2016) which consists of approximately 500k training queries. We use a blend of BM25 (Robertson et al., 1995) and CoCondenser (Gao and Callan, 2022) hard negatives which is publicly available[2].

For evaluation, we evaluate in-domain retrieval performance on the official MS MARCO passage retrieval DEV set of 6,980 queries. We also evaluate on TREC DL19 and DL20 that consist of 43 and 54 queries respectively, with in-depth annotation. We adopt the following official evaluation metrics: MRR@10 and Recall@1000 for DEV, and NDCG@10 for DL19 and DL20. We also include the BEIR dataset (Thakur et al., 2021) for out-of-domain evaluation. We evaluate the official evaluation metric NDCG@10 on 13 publically available testsets in the BEIR collection. We defer more dataset details to Appendix B.

4.2 Compared Methods

We include baseline methods of both dense retrieval and sparse retrieval. For dense retrieval, we include CoCondenser (Gao and Callan, 2022), bi-SimLM (Wang et al., 2023), SGPT (Muennighoff, 2022) and RepLlama (Ma et al., 2024). RepLlama is the closest dense retrieval baseline to our method as we use the same contrastive training objective and trainset. We include results from the original paper with Llama-2-7b (Touvron et al., 2023), as well as Llama-3.1-8b (Grattafiori et al., 2024) results from Zhuang et al. (2024)[3].

For sparse retrieval, we include the classical BM25 method (Robertson et al., 1995). We also include SPLADE++ SelfDistil as well as the latest SPLADE-v3 variants that use complex hard negative mining and self-distillation training strategy, reported by Formal et al. (2021a); Lassance et al. (2024). Lastly, we include SparseEmbed (Kong et al., 2023) as another competitive SPLADE variant. For all the baselines, we use results reported in their corresponding papers.

[2]`https://huggingface.co/datasets/Tevatron/msmarco-passage-aug`

[3]Note we skip RepLlama-3-8B's results on BEIR as they are not reported by Zhuang et al. (2024).

4.3 Implementation and Hyperparameters

As mentioned in Section 3.2, we examine the effectiveness of the proposed method using pre-trained Llama-3 family models as the retriever backbone. We use 1B and 8B model sizes: Llama-3.2-1B and Llama-3.1-8B[4].

Our implementation is based on PyTorch, Huggingface (Wolf et al., 2019), Tevatron (Gao et al., 2022) and Pyserini (Lin et al., 2021)'s Lucene integration. After the model is trained, we use Lucene to build an inverted index and do subsequent retrieval. For adaptation training, we adapt the pre-trained CLMs on MS MARCO passage corpus for 10K steps, with 2,048 sequence length and 32 global batch size. We employ a cosine learning rate scheduling with 1k steps warmup. We use sequence packing (Raffel et al., 2020) technique for computational efficiency. For contrastive training, we use LoRA fine-tuning (Hu et al., 2021) to balance in-domain and out-of-domain performance (Biderman et al., 2024). We use a similar training setup as RepLlama, i.e., 15 hard negatives per positive query-passage pair, together with in-batch negatives. We use 511 in-batch negatives for both 1B and 8B models, implying a global batch size of 32. We use techniques including Flash Attention 2 (Dao, 2023), gradient checkpointing, gradient accumulation and PyTorch FSDP (Zhao et al., 2023) for scalable training. We train 1B model for 3 epochs and 8B model for 1 epoch using cosine learning rate scheduling. We mainly tune four hyperparameters: learning rate, `FLOPs` regularization coefficients λ_Q and λ_D, and LoRA rank R. More hyperparameter details and hardware information are deferred to Appendix C. At the inference time, we merge the LoRA adapter to the backbone model, and use `bfloat16` precision for inference.

5 Results and Analysis

We discuss in-domain evaluation results (Section 5.1) and out-of-domain results (Section 5.2). We detail the setup and results of quantization evaluation (Section 5.3) and finally discuss unsuccessful attempts (Section 5.4).

Table 1: Performance and index size for MS MARCO in-domain evaluation. We highlight the highest performance within each section except for the index size column.

Model	Size	Dev MRR@10	Recall@1k	DL19 NDCG@10	DL20 NDCG@10	Index Size GB (↓)
Dense Retrieval						
CoCondenser (Gao and Callan, 2022)	110M	38.2	98.4	71.7	68.4	25
bi-SimLM (Wang et al., 2023)	110M	39.1	98.6	69.8	69.2	25
GTR-XXL (Ni et al., 2022)	4.8B	38.8	99.0	-	-	25
RepLlama-2-7B (Ma et al., 2024)	7B	41.2	99.4	74.3	72.1	135
RepLlama-3-8B (Zhuang et al., 2024)	8B	**42.8**	-	**74.5**	**73.9**	135
Sparse Retrieval						
BM25 ($k_1 = 0.9, b = 0.4$)	-	18.4	85.4	50.6	48.0	2.6
SPLADE++ (Formal et al., 2021a)	110M	37.8	98.5	**73.6**	72.8	2.6
SparseEmbed (Kong et al., 2023)	110M	39.2	98.1	-	-	-
SPLADE-v3 (Lassance et al., 2024)	110M	**40.2**	-	72.3	**75.4**	3.1
Proposed Method						
CSPLADE-1B	1.3B	38.2	98.5	73.2	68.9	2.6
CSPLADE-Echo-1B	1.3B	38.8	98.9	72.9	69.5	4.6
CSPLADE-Bi-1B	1.3B	**40.4**	**99.0**	**73.8**	**69.8**	5.6
CSPLADE-8B	8B	39.5	99.0	73.0	68.0	7.5
CSPLADE-Echo-8B	8B	40.8	98.9	73.5	70.7	4.5
CSPLADE-Bi-8B	8B	**41.3**	**99.1**	**74.1**	**72.8**	6.7

5.1 In-domain Retrieval

We show the in-domain results in Table 1. For dense retrieval baselines, we find RepLlama-3-8B improves upon RepLlama-2-7B, suggesting the backbone language model's capacity is critical for retrieval performance. In terms of sparse retrieval baselines, SPLADE-v3 achieves the highest performance, surpassing RepLlama-3-8B on the DL20 benchmark. This observation suggests knowledge distillation from a strong cross-encoder teacher is effective for improving BERT-sized model's performance.

For the proposed method, we find that CSPLADE-Echo and CSPLADE-Bi significantly outperforms CSPLADE variant. For example, CSPLADE-Bi-1B achieves 40.4 MRR@10 on DEV compared to CSPLADE-1b. This suggests the importance of enabling bidirectioanl information. In addition, we observe that 8B models outperform the 1B counterparts, reiterating the importance of backbone model capacity. Between CSPLADE-Echo and CSPLADE-Bi, CSPLADE-Bi performs slightly better. Although we note it also requires more careful hyperparameter tuning to achieve strong performance. Finally, we note that the best performing CSPLADE-Bi-8B still lags behind its dense retrieval counterpart. We hypothesize this performance gap stems from is RepLlama-3-8B's use of a single 4,096-dimension dense vector, which offers greater representational capacity compared to CSPLADE's sparse representation. We intend to control the size of inverted index for efficient retrieval. A more comprehensive investigation into the trade-off between retrieval effectiveness and index size is left for future work.

5.2 Zero-shot Retrieval

We show the zero-shot retrieval performance in Table 2. Due to space constraints, we report results only for SPLADE-v3 and RepLlama-2 as representative baselines, and refer readers to Appendix D for the full set of baseline comparisons.

We find that RepLlama-2-7b achieves better out-of-domain performance compared to other dense retrieval baselines (avg. 55.1 NDCG@10), and is also better compared to the most competitive sparse retrieval method SPLADE-v3. Meanwhile, CSPLADE-Echo-1B and CSPLADE-Bi-1B achieve average NDCG@10 scores of 49.5 and 49.4, respectively, underperforming SPLADE-v3. This again suggests the effectiveness of distillation training especially when backbone LM's capacity is limited. However, when we increase the capacity of backbone language model, the performance significantly improved. For example, CSPLADE-Echo-8B and CSplade-bi-8b outperform SPLADE-v3 by a large margin, and are able to achieve performance on par with RepLlama-2-7b. This observation suggests that the proposed method's effectiveness is also gener-

[4]https://huggingface.co/meta-llama

Table 2: Results for zero-shot passage retrieval evaluation. We highlight the best performance within each section.

Dataset	SPLADE-v3 110M	RepLlama-2 7B	CSPLADE-Echo-1B 1.3B	CSPLADE-Bi-1B 1.3B	CSPLADE-Echo-8B 8B	CSPLADE-Bi-8b 8B
Arguana	**50.9**	48.6	46.7	45.0	48.1	**48.9**
Climate-FEVER	23.3	**31.0**	23.8	21.8	**29.5**	29.4
DBPedia	**45.0**	43.7	39.5	39.0	**45.2**	44.5
FEVER	79.6	**83.4**	71.3	74.8	85.2	**86.5**
FiQA	37.4	**45.8**	35.0	36.3	39.9	**40.5**
HotpotQA	**69.2**	68.5	61.0	62.4	69.4	**69.8**
NFCorpus	35.7	**37.8**	33.2	32.4	**37.7**	37.2
NQ	58.6	**62.4**	54.5	55.4	59.8	**60.9**
Quora	81.4	**86.8**	81.5	79.6	86.9	**87.1**
SCIDOCS	15.8	**18.1**	16.0	15.1	17.4	**17.6**
SciFact	71.0	**75.6**	71.1	71.1	73.2	**73.9**
TREC-COVID	74.8	**84.7**	77.7	71.6	**84.0**	83.2
Touche-2020	29.3	**30.5**	32.1	37.7	38.5	**38.9**
Average	51.7	**55.1**	49.5	49.4	55.0	**55.3**

alizable to out-of-domain zero-shot retrieval.

5.3 Model Quantization and Latency

One big obstacle to applying LLM for real-world retrieval applications is latency. Therefore, we examine the efficacy of the prevalent inference optimization technique — quantization — in the case of learned sparse retrieval. We include two calibration-free quantization methods: LLM.int8 (Dettmers et al., 2022) and native PyTorch quantization implementation torchao (torchao team, 2024). For LLM.int8, we use `INT4` and `INT8` weight-only quantization in BitsAndBytes[5]. For torchao, we use `INT4` and `INT8` weight-only quantization and `INT8` weight-and-activation quantization implementation[6]. See Appendix E for more details about these quantization methods.

For quantization methods and the `bfloat16` baseline, we build the inverted index using larger batch size with quantized models. Then we benchmark latency by measuring the model's speed to encode queries by setting batch size to 1. We report retrieval performance MRR@10 on MS MARCO DEV set, and use queries/second as the latency metric[7].

We show the quantization evaluation results in Figure 1. First we notice that low bits quantization hurts the model's performance, e.g., 4-bit quantized models show significant performance degradation, while for 8-bit quantized models the degradation is less severe. The observation is consistent with existing works in quantization evaluation (Dettmers and Zettlemoyer, 2023; Xu et al., 2024a; Hong et al., 2024, *inter alia*). In terms of the inference speed, we find that at 1B scale, both quantized models actually slow down inference speed without customized GPU kernels support, compared to extensively optimized `bfloat16` baseline with Flash Attention 2 and `torch.compile`, although they do require less GPU memory. We also conduct experiments on quantization methods that require calibration, including GPTQ (Frantar et al., 2023) and AWQ (Lin et al., 2024), but find they lead to severe performance degradation, i.e., <10 MRR@10. The reason is the mismatch between the causal language modeling objective used in calibration and ranking objective used to fine-tune the retrieval model. To summarize, our findings highlight the need for a more careful and targeted investigation of quantization methods as well as developing efficient quantized model inference kernels tailored to retrieval models.

5.4 Unsuccessful Attempts

We discuss two unsuccessful attempts throughout the experiments of adapting pre-trained causal language model for learned sparse retrieval.

To mitigate the dying ReLU problem, we experimented with a biased reparameterization trick (Wang et al., 2024b) that is inspired by Gumbel-softmax trick (Jang et al., 2016). Here the trick is illustrated in PyTorch style:

```
z = F.relu(z).detach() + F.gelu(z)
                  - F.gelu(z).detach()
```

where `z.detach()` eliminates the gradients of `z`.

[5] `https://github.com/bitsandbytes-foundation/bitsandbytes`

[6] `https://github.com/pytorch/ao`

[7] We opt to not include retrieval time from Lucene index but solely focus on encoding speed from the backbone language model, as Lucene retrieval speed is the same for all quantization methods, and is mainly dependent on the index size and CPU hardware capacity.

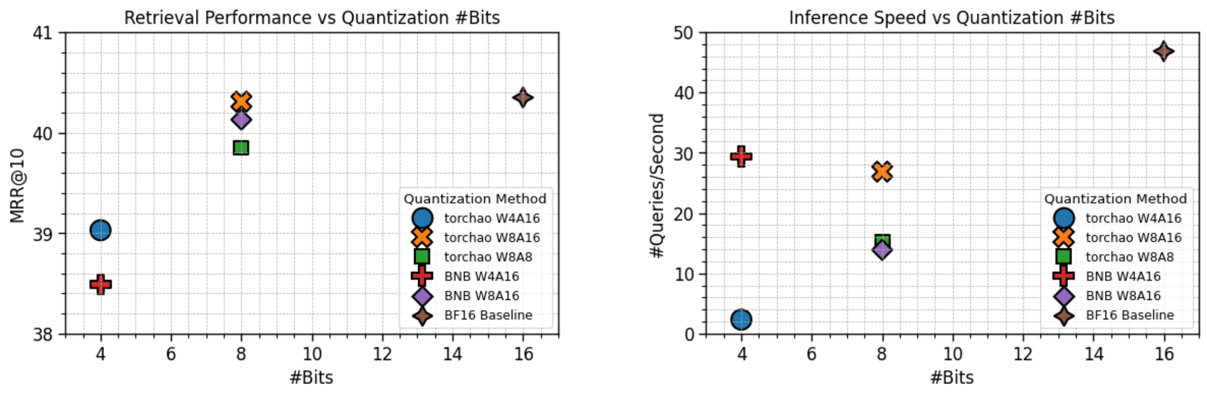

Figure 1: Quantization evaluation results for CSPLADE-Bi-1B. Left figure shows performance while right figure shows inference speed. See Appendix E for CSPLADE-Bi-8B results.

With this trick, the forward output still equals `F.relu(z)` but the gradient is computed with respect to `F.gelu(z)` which has gradients to negative input (Hendrycks and Gimpel, 2016). The reparameterization is biased as the gradient of GeLU is different from ReLU, which may lead to inferior performance. However, we found this trick do not fully mitigate training instability – the training still fails in certain combinations of hyperparameters.

In contrastive training phase, we have also experimented with using full parameter fine-tuning for Llama-3 instead of LoRA fine-tuning . We observe this leads to inferior results on BEIR datasets, while the similar problem is less significant for BERT-scale models. We hypothesize the reason is modern causal LLMs like Llama's extensive pretraining make them prone to overfitting. We leave this overfitting problem to future investigation.

6 Related Works

In this section, we discuss existing learned sparse retrieval methods, and refer to existing survey works for dense retrieval methods (Lin et al., 2022; Zhu et al., 2023b; Xu et al., 2025b). Zamani et al. (2018) propose SNRM to embed documents and queries in a sparse high-dimensional latent space, and enforce sparsity via l_1 regularization. DeepCT (Dai and Callan, 2019) learns to reweight terms via learning contextualized representations; but this method does not mitigate vocabulary mismatch problem as it does not employ query and document expansions. Later works further improve retrieval performance via expansion technique and corresponding aggregation mechanism, exemplified by SparTerm (Bai et al., 2020), SPARTA (Zhao et al., 2020) and EPIC (MacAvaney et al., 2020). SPLADE (Formal et al., 2021b) note pre-trained masked language model's masked language modeling head is particularly suited for projecting contextualized representations to vocabulary space. It additionally draws inspiration from $\log(\text{tf})$ (Fang et al., 2004) and employs `FLOPs` (Paria et al., 2020) regularization to improve performance. Later iterations of SPLADE (Formal et al., 2021a, 2022; Lassance et al., 2024) further improve by switching to MaxPooling aggregation mechanism and using sophisticated training strategies including hard negatives mining and distillation from cross-encoder teachers. Followup works propose to enable fine-grained query-document terms interactions to better capture relevance, examplified by SparseEmbed (Kong et al., 2023), SLIM (Li et al., 2023) and SPLATE (Formal et al., 2024). However, these studies are still limited to pre-trained masked language models, especially BERT (Devlin et al., 2019) family models. Some concurrent works have attempted to train SPLADE-style models beyond encoder backbones. Qiao et al. (2025) compare encoder-decoder models like Flan-T5 (Chung et al., 2024) versus decoder-only OPT (Zhang et al., 2022). But their experiments are limited to 3B-scale models, and do not include stronger pre-trained LLMs. Mistral-SPLADE (Doshi et al., 2024) train Mistral-7B (Jiang et al., 2023) with echo embeddings (Springer et al., 2024). In this work we formally examine the challenges of training learned sparse retrieval models with pre-trained LLMs as backbones, and propose corresponding mitigation strategies.

7 Conclusion and Future Works

In this paper, we focus on extending learned sparse retrieval, specifically SPLADE, to pre-trained causal language models. We identify two challenges: the training instability problem and the unidirectional information problem. To solve these problems, we propose a lightweight adaptation training phase to eliminate training instability and design two model variants to enable bidirectional information. With these techniques, we achieve competitive performance by training SPLADE with 8B scale pre-trained causal language model, while maintaining a minimized index size. Further, we analyze how model quantization affects learned sparse retrieval and discuss implications for future improvement.

We envision two future work directions. From the training perspective, the effect of dataset scaling, in both unsupervised adaption phrase training, and subsequent supervised fine-tuning should be carefully studied (Hoffmann et al., 2022). Further, different training strategies, such as knowledge distillation (Hinton et al., 2015) and Matryoshka Representation Learning (Kusupati et al., 2022) remain to be explored. From the inference perspective, the inference latency of casual LLMs as the retriever backbone is still prohibitive, and inference-free learned sparse retrieval (Formal et al., 2021b,a; Geng et al., 2025) is a promising future direction. How to further optimize retrieval index specifically for learned sparse retrieval is another important question (Bruch et al., 2024).

Limitations

In this work we focus on studying causal language models for learned sparse retrieval, specifically SPLADE due to its high performance. We benchmarked Llama-3 family models. Whether the proposed methodology applies to other backbone language model and learned sparse retrieval methods requires further investigation and benchmarking. The effectiveness of learned sparse retrieval on long documents should also be carefully examined. Given the limited space, we leave further investigation of model quantization to future work.

Ethical Considerations

This paper focuses on modeling improvement and the experiments are conducted on public benchmarks. To the best of our knowledge, this paper does not raise potential ethical concerns or risks.

References

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 preprint arXiv:2010.00768*.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. 2024. Llm2vec: Large language models are secretly powerful text encoders. *arXiv preprint arXiv:2404.05961*.

Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, et al. 2024. Lora learns less and forgets less. *Transactions on Machine Learning Research*.

Alexander Bondarenko, Maik Fröbe, Meriem Beloucif, Lukas Gienapp, Yamen Ajjour, Alexander Panchenko, Chris Biemann, Benno Stein, Henning Wachsmuth, Martin Potthast, et al. 2020. Overview of touché 2020: argument retrieval. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction: 11th International Conference of the CLEF Association, CLEF 2020, Thessaloniki, Greece, September 22–25, 2020, Proceedings 11*, pages 384–395. Springer.

Vera Boteva, Demian Gholipour, Artem Sokolov, and Stefan Riezler. 2016. A full-text learning to rank dataset for medical information retrieval. In *Advances in Information Retrieval: 38th European Conference on IR Research, ECIR 2016, Padua, Italy, March 20–23, 2016. Proceedings 38*, pages 716–722. Springer.

Sebastian Bruch, Franco Maria Nardini, Cosimo Rulli, and Rossano Venturini. 2024. Efficient inverted indexes for approximate retrieval over learned sparse representations. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 152–162.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S Weld. 2020. Specter: Document-level representation learning using citation-informed transformers. *arXiv preprint arXiv:2004.07180*.

Zhuyun Dai and Jamie Callan. 2019. Deeper text understanding for ir with contextual neural language modeling. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 985–988.

Tri Dao. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in neural information processing systems*, 35:30318–30332.

Tim Dettmers and Luke Zettlemoyer. 2023. The case for 4-bit precision: k-bit inference scaling laws. In *International Conference on Machine Learning*, pages 7750–7774. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Thomas Diggelmann, Jordan Boyd-Graber, Jannis Bulian, Massimiliano Ciaramita, and Markus Leippold. 2020. Climate-fever: A dataset for verification of real-world climate claims. *arXiv preprint arXiv:2012.00614*.

Meet Doshi, Vishwajeet Kumar, Rudra Murthy, Vignesh P, and Jaydeep Sen. 2024. Mistral-splade: Llms for better learned sparse retrieval.

Hui Fang, Tao Tao, and ChengXiang Zhai. 2004. A formal study of information retrieval heuristics. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 49–56.

Yan Fang, Jingtao Zhan, Qingyao Ai, Jiaxin Mao, Weihang Su, Jia Chen, and Yiqun Liu. 2024. Scaling laws for dense retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 1339–1349, New York, NY, USA. Association for Computing Machinery.

Thibault Formal, Stéphane Clinchant, Hervé Déjean, and Carlos Lassance. 2024. Splate: Sparse late interaction retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2635–2640.

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, page 2353–2359, New York, NY, USA. Association for Computing Machinery.

Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 2021a. Splade v2: Sparse lexical and expansion model for information retrieval.

Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021b. *SPLADE: Sparse Lexical and Expansion Model for First Stage Ranking*, page 2288–2292. Association for Computing Machinery, New York, NY, USA.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2023. OPTQ: Accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*.

Luyu Gao and Jamie Callan. 2022. 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)*, pages 2843–2853, Dublin, Ireland. Association for Computational Linguistics.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2022. Tevatron: An efficient and flexible toolkit for dense retrieval. *arXiv preprint arXiv:2203.05765*.

Zhichao Geng, Yiwen Wang, Dongyu Ru, and Yang Yang. 2025. Towards competitive search relevance for inference-free learned sparse retrievers.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Faegheh Hasibi, Fedor Nikolaev, Chenyan Xiong, Krisztian Balog, Svein Erik Bratsberg, Alexander Kotov, and Jamie Callan. 2017. Dbpedia-entity v2: a test collection for entity search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1265–1268.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Sebastian Hofstätter, Sophia Althammer, Michael Schröder, Mete Sertkan, and Allan Hanbury. 2020. Improving efficient neural ranking models with cross-architecture knowledge distillation. *arXiv preprint arXiv:2010.02666*.

Junyuan Hong, Jinhao Duan, Chenhui Zhang, Zhangheng Li, Chulin Xie, Kelsey Lieberman, James Diffenderfer, Brian R Bartoldson, Ajay Kumar Jaiswal, Kaidi Xu, et al. 2024. Decoding compressed trust: Scrutinizing the trustworthiness of efficient llms under compression. In *International Conference on Machine Learning*, pages 18611–18633. PMLR.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. arxiv 2021. *arXiv preprint arXiv:2106.09685*.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Albert Qiaochu Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, L'elio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *ArXiv*, abs/2310.06825.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

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)*, pages 6769–6781, Online. Association for Computational Linguistics.

Weize Kong, Jeffrey M. Dudek, Cheng Li, Mingyang Zhang, and Michael Bendersky. 2023. Sparseembed: Learning sparse lexical representations with contextual embeddings for retrieval. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 2399–2403, New York, NY, USA. Association for Computing Machinery.

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, et al. 2022. Matryoshka representation learning. *Advances in Neural Information Processing Systems*, 35:30233–30249.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Carlos Lassance, Hervé Déjean, Thibault Formal, and Stéphane Clinchant. 2024. Splade-v3: New baselines for splade. *arXiv preprint arXiv:2403.06789*.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2024. Nv-embed: Improved techniques for training llms as generalist embedding models. *arXiv preprint arXiv:2405.17428*.

Minghan Li, Sheng-Chieh Lin, Xueguang Ma, and Jimmy Lin. 2023. Slim: Sparsified late interaction for multi-vector retrieval with inverted indexes. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1954–1959.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. In *Proceedings of Machine Learning and Systems*, volume 6, pages 87–100.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: An easy-to-use python toolkit to support replicable ir research with sparse and dense representations. *arXiv preprint arXiv:2102.10073*.

Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. 2022. *Pretrained transformers for text ranking: Bert and beyond*. Springer Nature.

Lu Lu, Yeonjong Shin, Yanhui Su, and George Em Karniadakis. 2019. Dying relu and initialization: Theory and numerical examples. *arXiv preprint arXiv:1903.06733*.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2024. Fine-tuning llama for multi-stage text retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2421–2425.

Sean MacAvaney, Franco Maria Nardini, Raffaele Perego, Nicola Tonellotto, Nazli Goharian, and Ophir Frieder. 2020. Expansion via prediction of importance with contextualization. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 1573–1576.

Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. 2018. Www'18 open challenge: financial opinion mining and question answering. In *Companion proceedings of the the web conference 2018*, pages 1941–1942.

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*, pages 1723–1727.

Niklas Muennighoff. 2022. Sgpt: Gpt sentence embeddings for semantic search. *arXiv preprint arXiv:2202.08904*.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernandez Abrego, Ji Ma, Vincent Zhao, Yi Luan, Keith Hall, Ming-Wei Chang, and Yinfei Yang. 2022. Large dual encoders are generalizable retrievers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9844–9855, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Biswajit Paria, Chih-Kuan Yeh, Ian EH Yen, Ning Xu, Pradeep Ravikumar, and Barnabás Póczos. 2020. Minimizing flops to learn efficient sparse representations. In *International Conference on Learning Representations*.

Jacob Portes, Alexander Trott, Sam Havens, Daniel King, Abhinav Venigalla, Moin Nadeem, Nikhil Sardana, Daya Khudia, and Jonathan Frankle. 2023. Mosaicbert: A bidirectional encoder optimized for fast pretraining. *Advances in Neural Information Processing Systems*, 36:3106–3130.

Jingfen Qiao, Thong Nguyen, Evangelos Kanoulas, and Andrew Yates. 2025. Leveraging decoder architectures for learned sparse retrieval. *arXiv preprint arXiv:2504.18151*.

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*, pages 5835–5847, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at trec-3. *Nist Special Publication Sp*, 109:109.

Gerard Salton. 1984. The use of extended boolean logic in information retrieval. In *Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data*, SIGMOD '84, page 277–285, New York, NY, USA. Association for Computing Machinery.

Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

Jacob Mitchell Springer, Suhas Kotha, Daniel Fried, Graham Neubig, and Aditi Raghunathan. 2024. Repetition improves language model embeddings. *arXiv preprint arXiv:2402.15449*.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*.

torchao team. 2024. torchao: Pytorch native quantization and sparsity for training and inference.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ellen Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, William R Hersh, Kyle Lo, Kirk Roberts, Ian Soboroff, and Lucy Lu Wang. 2021. Trec-covid: constructing a pandemic information retrieval test collection. In *ACM SIGIR Forum*, volume 54, pages 1–12. ACM New York, NY, USA.

Henning Wachsmuth, Shahbaz Syed, and Benno Stein. 2018. Retrieval of the best counterargument without prior topic knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 241–251.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. *arXiv preprint arXiv:2004.14974*.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2023. SimLM: Pre-training with representation bottleneck for dense passage retrieval. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2244–2258, Toronto, Canada. Association for Computational Linguistics.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024a. Improving text embeddings with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11897–11916, Bangkok, Thailand. Association for Computational Linguistics.

Yifei Wang, Qi Zhang, Yaoyu Guo, and Yisen Wang. 2024b. Non-negative contrastive learning. In *The Twelfth International Conference on Learning Representations*.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, et al. 2024. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. *arXiv preprint arXiv:2412.13663*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*.

Zhichao Xu. 2023. Context-aware decoding reduces hallucination in query-focused summarization.

Zhichao Xu. 2024. Rankmamba: Benchmarking mamba's document ranking performance in the era of transformers.

Zhichao Xu and Daniel Cohen. 2023. A lightweight constrained generation alternative for query-focused summarization. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1745–1749.

Zhichao Xu, Ashim Gupta, Tao Li, Oliver Bentham, and Vivek Srikumar. 2024a. Beyond perplexity: Multi-dimensional safety evaluation of LLM compression. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 15359–15396, Miami, Florida, USA. Association for Computational Linguistics.

Zhichao Xu, Zhiqi Huang, Shengyao Zhuang, and Vivek Srikumar. 2025a. Distillation versus contrastive learning: How to train your rerankers.

Zhichao Xu, Hemank Lamba, Qingyao Ai, Joel Tetreault, and Alex Jaimes. 2024b. Cfe2: Counterfactual editing for search result explanation. In *Proceedings of the 2024 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 145–155.

Zhichao Xu, Fengran Mo, Zhiqi Huang, Crystina Zhang, Puxuan Yu, Bei Wang, Jimmy Lin, and Vivek Srikumar. 2025b. A survey of model architectures in information retrieval.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Andrew Yates, Carlos Lassance, Sean MacAvaney, Thong Nguyen, and Yibin Lei. 2024. Neural lexical search with learned sparse retrieval. In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, pages 303–306.

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*, pages 497–506.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, et al. 2025. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.

Tiancheng Zhao, Xiaopeng Lu, and Kyusong Lee. 2020. Sparta: Efficient open-domain question answering via sparse transformer matching retrieval. *arXiv preprint arXiv:2009.13013*.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. 2023. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*.

Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Haonan Chen, Zheng Liu, Zhicheng Dou, and Ji-Rong Wen. 2023a. Large language models for information retrieval: A survey. *arXiv preprint arXiv:2308.07107*.

Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Haonan Chen, Zheng Liu, Zhicheng Dou, and Ji-Rong Wen. 2023b. Large language models for information retrieval: A survey. *arXiv preprint arXiv:2308.07107*.

Shengyao Zhuang, Xueguang Ma, Bevan Koopman, Jimmy Lin, and Guido Zuccon. 2024. PromptReps: Prompting large language models to generate dense and sparse representations for zero-shot document retrieval. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4375–4391, Miami, Florida, USA. Association for Computational Linguistics.

A Code of Loss Function in Adaptation Training

We show the pseudo code in Figure 2.

B Dataset Details

Four of the datasets we used in experiments (NFCorpus (Boteva et al., 2016), FiQA-2018 (Maia et al., 2018), Quora[8], Climate-Fever (Diggelmann et al., 2020)) do not report the dataset license in the paper or a repository. For the rest of the datasets, we list their licenses below:

- MS MARCO (Bajaj et al., 2016): MIT License for non-commercial research purposes.
- ArguAna (Wachsmuth et al., 2018): CC BY 4.0.
- DBPedia (Hasibi et al., 2017): CC BY-SA 3.0.
- FEVER (Thorne et al., 2018): CC BY-SA 3.0.
- HotpotQA (Yang et al., 2018): CC BY-SA 4.0.
- NQ (Kwiatkowski et al., 2019): CC BY-SA 3.0.
- SCIDOCS (Cohan et al., 2020): GNU General Public License v3.0.
- SciFact (Wadden et al., 2020): CC BY-NC 2.0.
- TREC-COVID (Voorhees et al., 2021): "Dataset License Agreement".
- Touche-2020 (Bondarenko et al., 2020): CC BY 4.0.

C Hyperparameter Details

We show details of hyperparameters in Table 3. We mainly tune four hyperparameters: learning rate, LoRA Rank, `FLOPs` regularizer coefficient λ_Q and λ_D. We notice that increasing LoRA Rank did not improve retrieval performance, but led to performance degradation on BEIR datasets, therefore we use rank=16 for 8B models. 1B models are trained on a single EC2 p4d.24xlarge intance with 8xA100 40GB GPUs, while 8B models are trained on two p4d.24xlarge instances.

D Additional Experiment Results

We show additional BEIR results in Table 4.

E Quantization Evaluation Details

E.1 Quantization Methods

We experimented with quantization methods that require calibration, including AWQ (Lin et al., 2024) and GPTQ (Frantar et al., 2023), but find they led to significant performance degradation due to the misaligned objectives in model fine-tuning and calibration. We opted to focus on calibration-free quantization methods.

E.2 Hardwares

The inference speed is measured on single A100 GPU with 40GB memory. We use `torch.compile` and Flash Attention 2 for the `bfloat16` baseline.

E.3 Results for 8B Models

We report the quantization results for 8B models at Figure 3. We notice that at 8B scale, performance degradation is less severe compared to 1B models, and torchao W8A16 quantization improves inference speed compared to `bfloat16` baseline.

[8] https://www.kaggle.com/c/quora-question-pairs

```python
# Input:
# x: input sequence of tokens, shape (batch_size, sequence_length)
# model: causal language model
# targets: target sequence, which is the same as input, shifted by 1
# loss_fn: cross-entropy loss function
# adapt_loss_factor: hyperparameter to control the balance between two terms

import torch

# Model predicts logits for each token in the sequence
logits = model(x) # (batch_size, sequence_length, vocabulary)
shift_logits = logits[:, :-1, :] # Shift the logits by 1
shift_targets = x[:, 1:] # Shift the input sequence to get the target sequence

# Compute the causal language modeling loss
# Note that the tensors need to be flattened to be used in torch CrossEntropyLoss
clm_loss = loss_fn(shift_logits.view(-1, vocab_size), shift_targets.view(-1))

adapt_logits = torch.log(1 + torch.relu(shift_logits))
adapt_loss = loss_fn(adapt_logits.view(-1, vocab_size), shift_targets.view(-1))

loss = clm_loss + adapt_loss_factor * adapt_loss

# Return the computed loss
return loss
```

Figure 2: Pseudo code for adaption phase training loss computation.

Table 3: Hyperparameters used in training CSPLADE. We use 5% of the total training steps for learning rate warmup. Global BZ denotes global batch size..

Model	LR	LR Scheduler	#Epochs	Global BZ	LoRA Rank	λ_Q	λ_D
CSPLADE-1B	5e-5	Cosine	3	32	64	0.003	0.003
CSPLADE-Echo-1B	5e-5	Cosine	3	32	64	0.003	0.003
CSPLADE-Bi-1B	5e-5	Cosine	3	32	64	0.003	0.003
CSPLADE-8B	1e-4	Cosine	1	32	16	0.03	0.03
CSPLADE-Echo-8B	1e-4	Cosine	1	32	16	0.03	0.03
CSPLADE-Bi-8B	1e-4	Cosine	1	32	16	0.03	0.03

Table 4: Additional results for zero-shot passage retrieval evaluation. We highlight the best performance within each section.

	BM25	SPLADE++	SparseEmbed	SGPT	CSPLADE-1B	CSPLADE-8B
Dataset	-	110M	110M	5.8B	1.3B	8B
Arguana	39.7	**52.5**	51.2	51.4	44.7	**45.2**
Climate-FEVER	16.5	23.0	21.8	**30.5**	19.5	**27.2**
DBPedia	31.8	43.6	**45.7**	39.9	39.2	**41.8**
FEVER	65.1	79.3	**79.6**	78.3	73.3	**82.3**
FiQA	23.6	34.8	33.5	**37.2**	33.2	**39.5**
HotpotQA	63.3	68.7	**69.7**	59.3	63.6	**66.3**
NFCorpus	32.2	34.8	34.1	**36.2**	**36.5**	35.7
NQ	30.6	53.7	**54.4**	52.4	52.9	**58.8**
Quora	78.9	83.4	**84.9**	84.6	81.0	**87.7**
SCIDOCS	14.9	15.9	16.0	**19.7**	15.8	**17.0**
SciFact	67.9	70.2	70.6	**74.7**	71.2	**72.2**
TREC-COVID	59.5	72.7	72.4	**87.3**	68.4	**79.2**
Touche-2020	**44.2**	24.5	27.3	25.4	34.8	**38.0**
Average	43.7	50.5	50.9	**52.1**	48.8	**53.1**

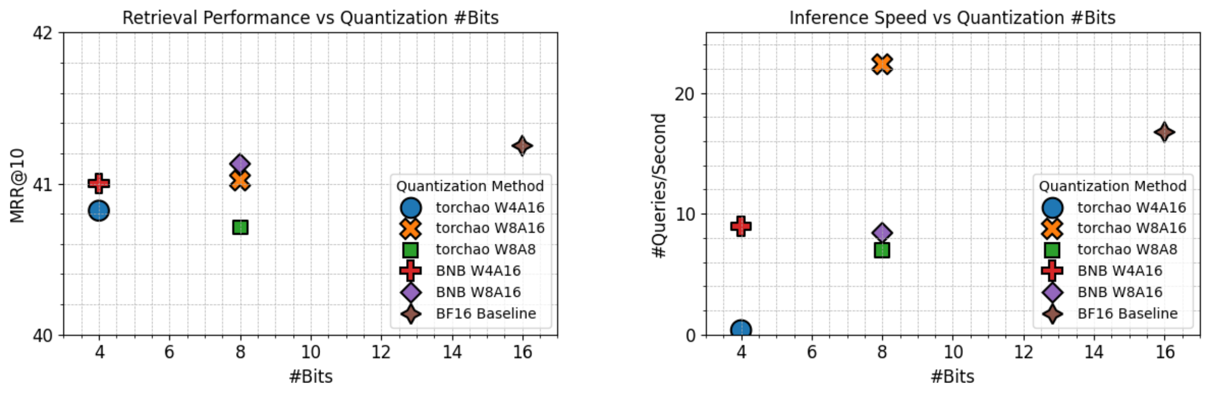

Figure 3: Quantization evaluation results for CSPLADE-Bi-8B. Left figure shows performance while right figure shows inference speed.