SEMI-PARAMETRIC RETRIEVAL VIA BINARY BAG-OF-TOKENS INDEX

Jiawei Zhou[1,3] **Li Dong**[2] **Furu Wei**[2] **Lei Chen**[1,3]
The Hong Kong University of Science and Technology[1] Microsoft Research[2]
The Hong Kong University of Science and Technology (Guangzhou)[3]
{jzhoubu,leichen}@ust.hk, {lidong1,fuwei}@microsoft.com

ABSTRACT

Information retrieval has transitioned from standalone systems into essential components across broader applications, with indexing efficiency, cost-effectiveness, and freshness becoming increasingly critical yet often overlooked. In this paper, we introduce **S**em**I**-parametric **D**isentangled **R**etrieval (SIDR), a bi-encoder retrieval framework that decouples retrieval index from neural parameters to enable efficient, low-cost, and parameter-agnostic indexing for emerging use cases. Specifically, in addition to using embeddings as indexes like existing neural retrieval methods, SIDR supports a non-parametric bag-of-tokens index for search, achieving BM25-like indexing complexity with significantly better effectiveness. Our comprehensive evaluation across 16 retrieval benchmarks demonstrates that SIDR outperforms both neural and term-based retrieval baselines under the same indexing workload: (i) When using an parametric embedding-based index, SIDR exceeds the performance of conventional neural retrievers while maintaining similar training complexity; (ii) When using a non-parametric tokenization-based index, SIDR matches the complexity of traditional term-based retrieval BM25, while consistently outperforming it on in-domain datasets; (iii) Additionally, we introduce a late parametric mechanism that matches BM25 index preparation time for search while outperforming both BM25 and other neural retrieval baselines in effectiveness. Code is available at https://github.com/jzhoubu/sidr.

1 INTRODUCTION

In recent years, information retrievers has evolved from end-to-end systems to essential components in various applications, including question answering (Kolomiyets & Moens, 2011; Zhu et al., 2021), classification (Long et al., 2022), recommendation (Dong et al., 2020; Manzoor & Jannach, 2022) and dialog systems (Liu et al., 2024b). This evolution has notably accelerated with the advent of the retrieval-augmented generation (RAG) paradigm (Bommasani et al., 2021; Guu et al., 2020; Yu et al., 2022; Mialon et al., 2023), in which the retrieval component enables large language models (LLMs) to access relevant data from external sources, effectively addressing challenges such as like hallucination (Ji et al., 2023; Zhang et al., 2023), obsolescence (Wang et al., 2023), and privacy concerns (Huang et al., 2022).

Traditional retrieval systems provide end-to-end search services and build indexes offline, without concern for cost or latency. In contrast, advanced retrieval components integrate with various downstream models, requiring greater flexibility to meet the diverse needs of applications. We present several **emerging retrieval scenarios** where current neural retrievers face limitations, and introduce specific ***index properties*** in our proposed framework designed to overcome these challenges.

Scenario 1: Online indexing for RAG applications with real-time knowledge sources. In RAG applications that depend on real-time knowledge source, such as up-to-the-minute internet information (Liu et al., 2023) and user-uploaded content requiring immediate responses (Wang et al., 2024), efficient ***online indexing*** is essential as it determines the time lag between the availability of data and its application. Effective neural retrieval with efficient online indexing facilitates the rapid assimilation and filtering of real-time datastreams, reducing computational burdens and mitigating hallucinations (Liu et al., 2024a; Shuster et al., 2021) when LLMs process long contexts.

Scenario 2: Low-cost index for exploration and deployment. Concerns about data privacy (Huang et al., 2022; Arora et al., 2023) and licensing issues (Min et al., 2023) are driving startups and individual developers towards building their own local retrieval pipelines for RAG applications. In this context, it is common to construct temporary retrieval indexes that are frequently modified and rebuilt to cater to different data sources and to optimize for varying chunk size selections, allowing for effective analysis of large datastores (Shao et al., 2024) and the fine-tuning of configurations. Given their sensitivity to resource constraints, smaller entities often prioritize cost reduction over maximizing effectiveness, making ***Low-cost Index*** more important than achieving state-of-the-art performance.

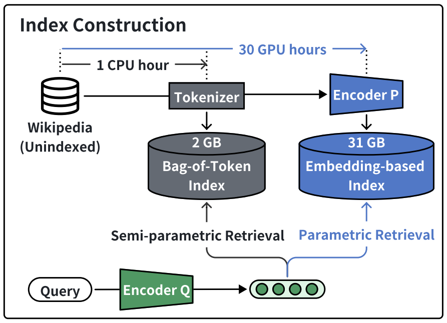

Figure 1: Comparison of storage (2 GB vs. 31 GB) and resource costs (1 CPU hour vs. 30 GPU hours) between two indexes.

Scenario 3: Parameter-agnostic index for co-training retrievers with LLMs. A significant challenge in co-training neural retrievers with LLMs is the index update issue (Asai et al., 2023) caused by in-training retrieval (Guu et al., 2020; Izacard et al., 2022a; Shi et al., 2023). Specifically, during training, a neural retriever parameterized by θ is learned to search information from a datastore $\mathcal{D}$ to enhance the LLMs. The retrieval index, denoted as $E_\theta(\mathcal{D})$, consists of the neural embeddings of the datastore $\mathcal{D}$. As the parameters update $\theta \rightarrow \theta'$, the datastore index needs to be rebuilt $E_\theta(\mathcal{D}) \rightarrow E_{\theta'}(\mathcal{D})$ to prevent it from becoming stale. This process is computationally expensive and compromises the training objectives. Developing a neural retrieval that supports a ***Parameter-agnostic Index*** could address this issue and streamline the co-training pipelines.

To meet these emerging needs, our paper introduces the **S**em**I**-parametric **D**isentangled **R**etrieval framework (SiDR), which decouples retrieval index from neural parameters to facilitate an *efficient*, *low-cost*, and *non-parametric* indexing setup. Specifically, our framework involves learning parametric term weighting within a language model vocabulary space, where non-parametric representations can be straightforwardly defined and constructed via tokenization. By aligning these two types of representations, one encoder within the bi-encoder framework can optionally utilize tokenization-based representations as a shortcut for indexing large data volumes. As a result, SiDR simultaneously supports a parametric index that utilizes neural embeddings and a non-parametric index that employs bag-of-tokens representations. As illustrated in Figure 1, using the non-parametric index for the Wikipedia corpus drastically reduces the indexing cost and time from 30 GPU hours to just 1 CPU hour and reduces storage size from 31GB to 2GB. This design offers flexibility in choosing indexes with varying complexity to meet diverse retrieval scenarios and co-training propose.

Our comprehensive evaluations across 16 retrieval benchmarks demonstrate that SiDR outperforms both neural and term-based retrieval baselines under comparable indexing workloads. Specifically, our framework with a non-parametric index achieves a 10.6% improvement in top-1 accuracy in-domain compared to BM25, while maintaining indexing efficiency on par with BM25. Additionally, when utilizing a parametric index, our framework surpasses neural retrieval methods by 2.7% with similar training complexity. Furthermore, our late parametric approach that retrieves from a non-parametric index and re-ranks the results on-the-fly, achieving indexing efficiency comparable to BM25 while maintaining the effectiveness of neural retrieval.

We summarize our contributions from two main aspects:

- From IR perspective: We introduce a versatile semi-parametric retrieval framework that supports both parametric and non-parametric indexes to accommodate diverse downstream scenarios. The parametric index uses neural embeddings for effectiveness, while the non-parametric relies solely on tokenization for efficiency. We further propose a late parametric mechanism to maximize the trade-off between retrieval effectiveness and indexing efficiency.

- From RAG perspective: Our approach introduces in-training retrieval on a fixed non-parametric index, which avoids index staleness and eliminates the need for costly index rebuilds within the retriever's training loop. This simplifies the co-training the retrieval system with other models.

2 Background

Information retrieval task. Given a query q and a datastore $\mathcal{D}$, information retrieval (Manning, 2009) aims to identify the most relevant passage $p \in \mathcal{D}$ based on q. This task is typically performed using a bi-encoder framework, which employs two independent encoders to embed queries and passages into vector representations. The retrieval process can be formulated as:

$$\hat{p} = \operatorname{argmax} f(q, \mathcal{D}) = \underset{\forall p \in \mathcal{D}}{\operatorname{argmax}} f(q, p)$$

In this equation, $\hat{p}$ is the retrieved passage, and f is a function measures the relevance between q and p, usually calculated as the inner product of their vector representations.

Notation. We use $E_\theta(\cdot)$ to denote a general neural embedding process, applicable to both dense and sparse retrieval. Specifically, for term-based and sparse lexical retrieval, a $|V|$-dimensional representation is used, where each dimension represents the weight of a token or word within vocabulary V. We denote this embedding function as $V_{\text{Model}_\theta}(\cdot) : x \to \mathcal{R}^{|V|}$, where the subscript indicates the model architecture, and θ reflects whether the embedding involves learnable parameters.

Traditional term-based retrieval. Traditional term-based retrieval, such as TF-IDF (Ramos et al., 2003) and BM25 (Robertson et al., 2009), assess relevance based on weighted term overlap, which can be described as:

$$f_{\text{BM25}}(q, p) = \langle V_{\text{BM25}}(q), V_{\text{BM25}}(p) \rangle = \langle w_{\text{BM25}} \cdot V_{\text{BoW}}(q), w_{\text{BM25}} \cdot V_{\text{BoW}}(p) \rangle$$

These methods do not involve learned neural parameters and are therefore categorized as non-parametric (Min et al., 2022; Freeman et al., 2002). They employ a million-scale dimensional bag-of-words (BoW) representation $V_{\text{BoW}}(\cdot)$, with heuristic statistical metrics determining the term weighting w_{BM25} for each dimension. Due to the efficiency and cost-effectiveness in constructing the term-based index $V_{\text{BM25}}(\mathcal{D})$, these methods are still widely used in industry applications.

Neural retrieval. Unlike term-based retrieval that is heuristic-driven, neural retrieval (Karpukhin et al., 2020; Zhu et al., 2023) is data-driven and parameterized, tailored to learn on specific datasets and tasks. The relevance assessment is defined as:

$$f_\theta(q, p) = \langle E_\theta(q), E_\theta(p) \rangle$$

While neural retrievers are effective with ample training data, the construction of the parametric index $E_\theta(\mathcal{D})$ requires embedding the entire datastore, which introduces significant computational costs and latency that hinder their widespread adoption.

Neural retrievers can be further categorized into entangled retrievers (a.k.a. dense retrievers) and disentangled retrievers (a.k.a. sparse lexical retrievers), categories which we detail in Appendix A. Specifically, entangled retrievers utilize latent representations with dimensions such as 768, 1024, or 2048. In contrast, disentangled retrievers employ disentangled representations, with dimensions equal to vocabulary sizes, such as 30522 for BERT's vocabulary, where each dimension reflects the importance of each token.

3 Methodology

As an overview, **Sem**i-parametric **D**isentangled **R**etrieval (SiDR) is a disentangled retrieval system that builds on the VDR architecture (Zhou et al., 2024), incorporating modifications in the learning objective and utilizing in-training retrieval for negative mining to enhance its effectiveness. At downstream, SiDR supports both embedding-based index and bag-of-tokens index. The primary goal of our work is to expand the functionality of the current retrieval system to better accommodate emerging scenarios.

In the following sections, we delve into details of representation (§3.1), rational (§3.2), training objectives (§3.3), search pipelines (§3.4), and the application of in-training retrieval techniques (§3.5).

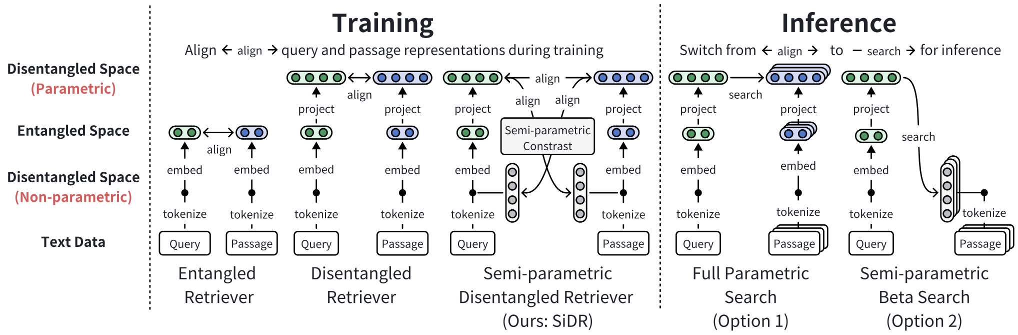
Training
Align ← align → query and passage representations during training
Inference
Switch from ← align → to − search → for inference
Disentangled Space (Parametric)
Entangled Space
Disentangled Space (Non-parametric)
Text Data
align
project
embed
tokenize
Query
Passage
Semi-parametric Contrast
search
Entangled Retriever
Disentangled Retriever
Semi-parametric Disentangled Retriever (Ours: SiDR)
Full Parametric Search (Option 1)
Semi-parametric Beta Search (Option 2)

Figure 2: Left: Training frameworks of entangled retriever, disentangled retriever and our proposed semi-parametric disentangled retriever SIDR; Right: Different inference pipelines of SIDR.

3.1 Parametric and Non-parametric Representation

Disentangled retrievers typically represent data in a language model vocabulary space, which can be interpreted as a set of tokens with weights. Our parametric representation, $V_\theta(x)$, uses token weights learned by a neural encoder, whereas the non-parametric representation, $V_{\text{BoT}}(x)$, can be viewed as using unweighted tokens generated by a tokenizer.

Parametric representation. We inherit the VDR encoder (Zhou et al., 2024), which extends the conventional MLMs architecture with three modification: (i) replacing the softmax activation with $elu1p$ to map dimensional values from $(0, 1)$ to $(0, +\infty)$; (ii) applying max-pooling to aggregate token representations into a global representation; and (iii) employing top-k sparsify (S_{topk}) to prune the less significant dimensional values to zero. These modifications can be expressed as follows:

$$elu1p(x) = \begin{cases} x+1 & \text{if } x >= 0 \\ e^x & \text{otherwise} \end{cases}$$

$$V_\theta(x) = \text{S}_{topk} \circ \text{MaxPool} \circ \{V_{\text{MLMH}_\theta + \text{elu1p}}(t_i|x), \forall t_i \in x\}$$

Significantly, these modifications aggregate token representations into a global one, while preserving the property of assigning larger values to more relevant dimensions.

Non-parametric representation. The non-parametric bag-of-tokens (BoT) representation for a sequence of tokens x is defined as follows:

$$V_{\text{BoT}}(x) = \text{MaxPool} \circ \{V_{\text{BoT}}(t_i), \forall t_i \in x\}; \quad V_{\text{BoT}}(x)[i] = \begin{cases} 1 & \text{if } V[i] \in x \\ 0 & \text{otherwise} \end{cases}$$

$V_{\text{BoT}}(x)$ can be seen as the result of applying max pooling to the one-hot representations of all tokens in x, assigning each i-th dimension a value of 1 or 0, depending on whether the i^{th} token in V is present in x. Compared to $V_{\text{BM25}}(\cdot)$, $V_{\text{BoT}}(\cdot)$ is tokenizer-specific, with dimensionality on the scale of tens of thousands, and uses binary values that require less storage space, making it well-suited for tensorization and efficient GPU computation.

3.2 Rationale of semi-parametric alignment.

In this section, we elaborate on the rationale behind the semi-parametric design. The semi-parametric alignment is designed to be consistent with mask language models (MLMs) (Devlin et al., 2018) pre-training objecitve.

Mask language model pre-training. During pre-training, MLMs are optimized to predict masked tokens by leveraging the context. Specifically, given an input sequence of tokens $x = [t_1, t_2, \ldots, M(t_i), \ldots, t_n]$ with t_i masked, the MLM uses its prediction head (MLMH) with a softmax function to produce the probability $V_{\text{MLMH}_\theta + \text{softmax}}(\text{Mask}(t_i)|x)$ of the masked token t_i over a vocabulary. The ground truth probability is the one-hot representation of the token t_i. In this paper, we refer to this type of representation as the bag-of-tokens (BoT) representation, denote as $V_{\text{BoT}}(t_i)$. The mask token prediction task can be viewed as alignment between the vocabulary distribution of the masked token position with the one-hot representation $V_{\text{BoT}}(t_i)$ in a masked setup:

$$V_{\text{MLMH}_\theta + \text{softmax}}(\text{Mask}(t_i)|x) \overset{\text{align}}{\longleftrightarrow} V_{\text{BoT}}(t_i) \tag{1}$$

As a result, the representation $V_{\text{MLMH}_\theta}(t_i|x)$ tends to assign large values to the dimension corresponding to t_i, or that are semantically related to t_i based on the context x.

Semi-parametric alignment. The alignment between parametric and non-parametric representation can be expressed as follows:

$$\text{S}_{topk} \circ \text{MaxPool} \circ \{V_{\text{MLMH}_\theta + \text{elu1p}}(t_i|x), \forall t_i \in x\} \overset{\text{align}}{\longleftrightarrow} \text{S}_{topk} \circ \text{MaxPool} \circ \{V_{\text{BoT}}(t_i), \forall t_i \in x\}$$

The semi-parametric alignment is modeled after the MLM pre-training objective, as detailed in Equation 1, and expands by aligning multi-token representations between the query and passage. The consistency between upstream pre-training and downstream tuning supports the alignability of these two representations.

3.3 Semi-parametric Contrastive Learning

In a batch containing N instances, each instance consists of a query q_i, a positive passage p_i, and a set of of negative passages. Our training objective is based on contrastive learning (Jaiswal et al., 2020), which aims to maximize the similarity of positive pairs $f(q_i, p_i)$ for all instances i, while minimize the similarity of all negative pairs, denoted as $f(q_i, p_j)$ for all $j \neq i$. The loss function is defined as follows:

$$L(q,p) = -\sum_{i=1}^{N}(\log \underbrace{\frac{e^{f(q_i,p_i)}}{\sum_{\forall p \in B} e^{f(q_i,p)}}}_{\text{q-to-p}} + \log \underbrace{\frac{e^{f(p_i,q_i)}}{\sum_{\forall q \in B} e^{f(p_i,q)}}}_{\text{p-to-q}})$$

This results in a final loss that integrates both parametric and semi-parametric components:

$$\begin{aligned} L_{\text{para}}(q,p) &= L(V_\theta(q), V_\theta(p)) \\ L_{\text{semi-para}}(q,p) &= L(V_\theta(q), V_{\text{BoT}}(p))/2 + L(V_{\text{BoT}}(q), V_\theta(p))/2 \\ L_{\text{final}}(q,p) &= L_{\text{para}}(q,p) + L_{\text{semi-para}}(q,p) \end{aligned}$$

The parametric contrastive loss L_p aims to align the parametric representations of q and p, a common objective for retrieval training. The semi-parametric contrastive loss L_{sp} ensures interaction between the non-parametric and parametric representations, which forms the foundation of our model to support a BoT index $V_{\text{BoT}}(\mathcal{D})$.

3.4 Search Pipelines and Index Types

Our framework supports both a parametric embedding-based index and a non-parametric tokenization-based index. Below, we discuss search functions and the corresponding index type.

Full parametric search (SiDR$_{\text{full}}$) utilizes a **parametric index** $V_\theta(\mathcal{D})$, which relies on embeddings derived from a neural encoder for the datastore. The relevance is defined as:

$$f_\theta(q, \mathcal{D}) = \langle V_\theta(q), V_\theta(\mathcal{D}) \rangle$$

This is the common indexing process for neural retrieval systems, which are effective but involve higher costs and longer latency for embedding the entire $\mathcal{D}$ to obtain the index $V_\theta(\mathcal{D})$.

Semi-parametric beta search (SiDR$_\beta$) leverages a **non-parametric index** $V_{\text{BoT}}(\mathcal{D})$ based on BoT representations of the datastore, which are constructed solely by a tokenizer. The relevance is defined as:

$$f_\beta(q, \mathcal{D}) = \langle V_\theta(q), V_{\text{BoT}}(\mathcal{D}) \rangle$$

Beta search applies BoT representations on the index side, eliminating the need for neural embeddings during index processing for large datastores, making it suitable for various applications.

Late parametric with top-m re-rank (SIDR$_\beta$ (m)) is a search pipeline that starts search with a non-parametric index to retrieve top-m passages, denote as $\mathcal{D}_m$, and then embeds them for re-ranking:

$$f_\beta(q, \mathcal{D}) = \langle V_\theta(q), V_{\text{BoT}}(\mathcal{D}) \rangle; \quad f_\theta(q, \mathcal{D}_m) = \langle V_\theta(q), V_\theta(\mathcal{D}_m) \rangle$$

Late parametric retrieval requires a first-stage retriever to support a non-parametric index, followed by a second-stage bi-encoder retriever to re-rank and cache the passage embeddings. From an efficiency and cost perspective, the late parametric with top-m re-rank only requires embedding at most $N_q \times m$ passages, where N_q is the number of queries. It starts the search with a non-parametric index, distributing the embedding workload throughout the online search process, achieving a fast retrieval setup. When dealing with exploratory scenarios that have limited queries or extremely large datastores, this approach becomes more efficient, as $N_q \times m$ remains much smaller than $|\mathcal{D}|$.

3.5 In-training Retrieval for Negative Sampling

While semi-parametric retrieval offers advantages for in-training retrieval by eliminating the need for re-indexing, it also has drawbacks, notably its limited effectiveness due to using non-parametric representations on index side. To enhance their performance, we integrate beta search in the training loop to dynamically source hard negative passages, leveraging the strengths of semi-parametric design to counterbalance their limitations. Specifically, during the training, SIDR employs beta search to retrieve the top-m passages in real-time — using $V_\theta(q)$ to search on non-parametric index $V_{\text{BoT}}(\mathcal{D})$ and get the top-m results $\mathcal{D}_m$. Subsequently, each passage in $\mathcal{D}_m$ is assessed whether it is negative based on exact matches with the answer strings. For each query, one negative is randomly selected from the identified negatives. This method is exclusively used for the Wikipedia benchmark, which provides answer strings to distinguish between negative and positive passages.

While in-training retrieval is increasingly adopted to enhance retrieval training (Zhan et al., 2021; Xiong et al., 2020) and facilitate co-training retrievers with LLMs (Shi et al., 2023), previous approaches have necessitated periodic index refreshment in the training loop. In contrast, our approach uniquely leverages a fixed index $V_{\text{BoT}}(\mathcal{D})$, eliminating the need for re-indexing $\mathcal{D}$. Our analysis shows that incorporating in-training retrieval with our non-parametric index does not add significant latency; however, a slight latency increase occurs due to the string matching process to identify negatives, which is discussed in Section 4.3.

4 Experimental Setup

4.1 Datasets

Wiki21m benchmark. Following established benchmark in retrieval literature (Chen et al., 2017; Karpukhin et al., 2020), we train our model on the training splits of Natural Questions (NQ; Kwiatkowski et al., 2019), TriviaQA (TQA; Joshi et al., 2017), and WebQuestions (WQ; Berant et al., 2013) datasets, and evaluated it on their respective test splits. The retrieval corpus used is Wikipedia, which contains over 21 million 100-word passages.

BEIR benchmark. We train our model on MS MARCO passage ranking dataset (Bajaj et al., 2016), which consists of approximately 8.8 million passages with around 500 thousand queries. The performance is assessed both in-domain on MS MARCO and in a zero-shot setting across 12 diverse datasets within the BEIR benchmark (Thakur et al., 2021).

4.2 Baselines

Primary baselines. We primarily compare our model with several established retrieval baselines, selected due to their similar model sizes and training complexities, ensuring a comparable training cost. These include dense retrieval DPR, term-based retrieval BM25, and sparse lexical retrieval models such as SPLADE (Formal et al., 2021b), uniCOIL (Lin & Ma, 2021), and VDR (Zhou et al., 2024). Specifically, VDR$_\beta$ refer to directly using the VDR models to perform beta search.

Advanced baselines. We also introduce advanced retrieval systems for baselines, including ANCE (Xiong et al., 2020), LexMAE (Shen et al., 2022), SPLADE-v2 (Formal et al., 2021a), Contriever (Izacard et al., 2021), GTR (Ni et al., 2021), E5 (Wang et al., 2022), and Dragon (Lin et al., 2023a). These systems leverage larger foundational models (Ma et al., 2023; Ni et al., 2021), retrieval-oriented pre-training (Fan et al., 2022; Zhou et al., 2022), or knowledge distillation (Formal et al., 2022) to enhance performance. We have categorized them as advanced baselines due to their significantly higher training costs associated with the additional training techniques. Future work may explore their integration with our model to assess potential benefits.

4.3 Implementation Details

Hyperparameters. For the NQ, TQA, and WQ datasets, our model is trained for 80 epochs, utilizing in-training retrieval for negative sampling. For the MS MARCO dataset, the training duration is set to 40 epochs. We utilize a batch size of 128 and an AdamW optimizer (Loshchilov & Hutter, 2018) with a learning rate set at 2×10^{-5}. Our model use a top-k sparsification with $k = 768$, matching the dimensionality of conventional dense retrieval embeddings. For computational devices, our systems are equipped with 4 NVIDIA A100 GPUs and Intel Xeon Platinum 8358 CPUs.

Training cost. The training durations for DPR, VDR, and SiDR on the NQ dataset are 5, 8, and 9 hours respectively, with 80 epochs under similar conditions. For SiDR, the training time per epoch without in-training retrieval is 6 minutes, which is identical to VDR. This duration increases to 8 minutes per epoch when incorporating in-training retrieval. The additional time is primarily due to the string matching process required to identify negative samples from the retrieved top-k passages.

5 Experiments

5.1 Main Results

Table 1: Top-1/5/20 retrieval accuracy on test sets (i.e., percentage of questions for which the answers is found in the retrieved passages). **Bold** numbers indicate the best performance within each setting.

	NQ			TQA			WQ		
	top1	top5	top20	top1	top5	top20	top1	top5	top20
Parametric Index									
DPR	46.0	68.9	80.2	54.1	71.5	80.0	37.4	59.7	**73.2**
VDR	43.8	68.0	79.9	52.9	71.3	79.3	37.1	58.7	72.5
SiDR$_{\text{full}}$	**49.1**	**69.3**	**80.7**	**56.2**	**73.0**	**80.5**	**40.2**	**61.0**	**73.2**
Non-parametric Index									
BM25	22.7	43.6	62.9	48.2	66.4	76.4	19.5	42.6	62.8
VDR$_\beta$	12.3	30.0	46.8	16.9	31.6	45.9	7.7	22.4	39.2
SiDR$_\beta$	**39.8**	**62.9**	**76.3**	**50.4**	**70.7**	**79.5**	**32.1**	**54.1**	**69.8**
Late parametric with top-m re-rank									
BM25+DPR ($m = 5$)	32.2	43.6	62.9	54.8	66.4	76.4	28.0	42.6	62.8
BM25+DPR ($m = 20$)	39.4	55.5	62.9	55.4	71.0	76.4	34.6	53.2	62.8
BM25+DPR ($m = 100$)	44.4	63.6	73.5	56.6	72.3	80.5	39.9	59.2	70.2
SiDR$_\beta$ ($m = 5$)	44.9	60.6	76.4	54.9	70.7	79.6	38.0	54.1	69.8
SiDR$_\beta$ ($m = 20$)	49.5	68.2	76.4	56.7	72.9	79.5	39.6	59.5	69.8
SiDR$_\beta$ ($m = 100$)	**50.3**	**70.7**	**80.6**	**56.8**	**73.3**	**81.3**	**41.5**	**62.0**	**73.5**

Wiki21m benchmark results. As shown in Table 1, when using a parametric index, SiDR$_{\text{full}}$ outperforms DPR and VDR in top-1 retrieval accuracy by 2.6% and 3.8%, respectively. These results demonstrate that although our primary objective is to enable neural retrieval with support for non-parametric indexing, our modifications do not diminish effectiveness with an embedding-based index; in fact, they may even improve it. This enhancement also suggests that current benchmarks may favor lexical relevance, likely due to their construction relying on term-based retrieval methods.

When utilizing a non-parametric index, SiDR$_\beta$ significantly surpasses VDR$_\beta$ and BM25 in top-1 accuracy by 28.5% and 10.6%, respectively. Unlike BM25, which relies on empirically derived heuristic term weights for indexing, SiDR$_\beta$ employs binary values in a significantly smaller vocabulary space, facilitating tensorization on GPUs. With ample in-domain training data, SiDR$_\beta$ significantly outperforms BM25, demonstrating the exceptional learnability and generalizability of neural retrievers in this context.

Additionally, late parametric methods improve SiDR$_\beta$ by enabling on-the-fly re-ranking of the top-m passages. Our results show that by re-ranking the top-20 passages, SiDR$_\beta$ $(m = 20)$ matches the performance of SiDR$_{\text{full}}$, and by extending re-ranking to the top-100 passages, it surpasses all primary baselines. Beyond effectiveness, the most significant advantage of late parametric methods is their substantial reduction in computational costs. For example, in evaluations on the NQ test split with 3k queries, SiDR$_{\text{full}}$ requires embedding the entire $\mathcal{D}$, which consists of 21 million passages. In contrast, SiDR$_\beta$ $(m = 100)$ only needs to embed $N_q \times m$ passages, amounting to just 1% of the passages in $\mathcal{D}$, yet achieves superior effectiveness. This underscores the exceptional suitability of late parametric methods for exploration or evaluation scenarios.

BEIR benchmark results. As shown in Table 2, SiDR$_{\text{full}}$ surpasses VDR, DPR, and other primary baselines in the BEIR benchmark when using either a parametric index or a non-parametric index with late parametric techniques, consistent with the findings from the Wiki21m benchmark. However, when relying solely on a non-parametric index, SiDR$_\beta$ outperforms VDR$_\beta$ and BM25 on in-domain datasets but falls behind BM25 on most out-of-domain datasets. We attribute this performance decline to three factors. First, due to the lack of answer strings to accurately identify negative passages in MS MARCO, we do not implement in-training retrieval during training on MS MARCO, which likely contributes to weaker performance. Second, as non-parametric indexes lack neural parameters, they are more sensitive to shifts in data distribution, which may lead to weaker effectiveness in out-of-domain scenarios. Lastly, many BEIR datasets exhibit a lexical bias due to their construction using BM25, as noted in the BEIR paper (Thakur et al., 2021), which inherently gives BM25 an advantage.

Table 2: Retrieval performance on MS MARCO (MRR@10) and BEIR benchmark (NDCG@10). **Bold** numbers indicate the best performance within each setting.

	MS MARCO	*ArguAna*	*Climate-FEVER*	*DBPedia*	*FEVER*	*FiQA*	*HotpotQA*	*NFCorpus*	*NQ*	*SCIDOCs*	*SciFact*	*TREC-COVID*	*Touché-2020*	Avg.
Advanced Retrieval Baselines														
LexMAE	**48.0**	50.0	21.9	42.4	**80.0**	35.2	**71.6**	34.7	56.2	15.9	71.7	76.3	**29.0**	**48.7**
Splade-v2	43.3	47.9	23.5	**43.5**	78.6	33.6	68.4	33.4	52.1	15.8	69.3	71.0	27.2	47.0
Contriever	-	44.6	23.7	41.3	75.8	32.9	63.8	32.8	49.8	16.5	67.7	59.6	23.0	44.3
GTR-base	42.0	51.1	**24.1**	34.7	66.0	34.9	53.5	30.8	49.5	14.9	60.0	53.9	20.5	41.2
E5-base	43.1	**51.4**	15.4	41.0	58.2	**36.4**	63.3	**36.1**	**62.9**	**19.0**	**73.1**	**79.6**	28.3	47.1
Dragon	39.3	48.9	22.2	41.7	78.1	35.6	64.8	32.9	53.1	15.4	67.5	74.0	24.9	46.6
ANCE	33.8	41.5	19.8	28.1	66.9	29.5	45.6	23.7	44.6	12.2	50.7	65.4	28.4	38.0
Parametric Index														
DPR	30.2	40.8	16.2	30.4	63.8	23.7	45.2	26.1	43.2	10.9	47.4	60.1	22.1	35.8
UniCOIL	32.9	35.5	15.0	30.2	72.3	27.0	64.0	32.5	36.2	13.9	**67.4**	59.7	25.9	39.4
SPLADE	34.0	43.9	**19.9**	36.6	73.0	28.7	63.6	31.3	46.9	14.5	62.8	67.3	20.1	42.4
VDR	**34.3**	48.6	17.6	39.0	**74.0**	28.8	**65.5**	**33.0**	47.2	**15.3**	67.3	67.8	**29.8**	44.5
SiDR$_{\text{full}}$	34.2	**53.0**	17.9	**39.3**	71.5	**29.8**	65.4	**33.0**	**47.7**	15.1	66.2	**68.0**	29.7	**44.7**
Non-parametric Index														
BM25	18.7	31.5	**21.3**	**31.3**	**75.3**	**23.6**	**60.3**	**32.5**	**32.9**	**15.8**	**66.5**	**65.6**	**36.7**	**41.1**
VDR$_\beta$	6.1	14.1	6.1	7.9	28.4	6.4	5.7	23.9	6.8	8.1	54.5	21.9	9.2	16.1
SiDR$_\beta$	**19.0**	**38.6**	10.8	20.8	46.5	19.8	49.4	27.9	25.3	11.1	64.2	53.5	23.7	32.5
Late parametric with top-m re-rank														
SiDR$_\beta$ $(m = 10)$	26.3	44.0	12.1	24.9	57.3	22.8	55.4	30.2	33.1	12.2	65.5	54.9	25.2	36.5
SiDR$_\beta$ $(m = 20)$	29.2	48.0	13.8	30.6	62.3	25.7	58.7	32.3	38.4	13.8	65.6	59.1	27.3	39.6
SiDR$_\beta$ $(m = 100)$	**32.9**	**51.5**	**16.6**	**37.8**	**69.2**	**29.3**	**63.3**	**33.2**	**44.7**	**14.8**	**65.7**	**65.9**	**29.0**	**43.4**

5.2 Retrieval Latency

We evaluated the latency of various retrieval systems across different stages using NQ test split and Wikipedia corpus, as shown in Table 3. The comparison assumes that both BM25 and SiDR indexes fit entirely into CPU/GPU memory. Further details can be found in Appendix B.

Indexing stage. The indexing stage converts the textual corpus into a searchable format. Both SiDR$_\beta$ and BM25 use tokenzation-based index and can complete indexing within 1 hour on a CPU, much faster than the over 20 hours required on GPUs for embedding-based index. The indexing stage often accounts for a large portion of the overall time and cost in the retrieval pipeline. Our BoT index is efficient, more cost-effective and benefiting from parallelization, making it a flexible option for practical retrieval-based applications.

Table 3: Latency at each stage of the retrieval pipeline. $\mathrm{T}(\cdot)$: computations of tokenization; $\mathrm{E}_\theta(\cdot)$: computation of neural model forward. †: Computations performed on a single GPU; times in parentheses indicate the latency if performed on a single CPU thread
. $\mathrm{E}_\theta(p)$ in search stage refers to the passage embedding used for late parametric re-ranking.

Model	Indexing		Search					Total
	$\mathrm{T}(\mathcal{D})$	$\mathrm{E}_\theta(\mathcal{D})$	$\mathrm{E}_\theta(q)$	$\mathrm{T}(q)$	$\mathrm{f}(q, \mathcal{D})$	$\mathrm{E}_\theta(p)$	Total	
BM25	0.6h	/	/	40s† (2m)		/	40s	0.6h
DPR	/	20.3h†	12s†(2m)	/	41ms†(2m)	/	12s	20.3h
VDR	/	23.7h†	15s†(2m)	/	130ms†(20m)	/	15s	23.7h
SIDR$_{\text{full}}$	/	23.7h†	15s†(2m)	/	130ms†(20m)	/	15s	23.7h
SIDR$_\beta$	0.5h	/	15s†(2m)	/	30ms† (3m)	/	15s	0.5h
SIDR$_\beta$ ($m = 20$)	0.5h	/	15s†(2m)	/	30ms†(3m)	4m†	4m	0.6h
SIDR$_\beta$ ($m = 100$)	0.5h	/	15s†(2m)	/	30ms†(3m)	20m†	20m	0.9h

Search stage. The search stage processes online incoming queries and retrieves relevant items from the indexed data. As shown in the table, SIDR$_\beta$ achieves significantly higher efficiency compared to BM25 and performs on par with dense retrieval methods when utilizing GPU resources. This advantage arises because the BoT index $V_{\text{BoT}}(\mathcal{D})$ has a fixed dimensionality, enabling tensorization for inner product calculations on the GPU. In contrast, BM25 term-based index $V_{\text{BM25}}(\mathcal{D})$ operates with millions of dimensions and relies on an inverted index for efficiency.

6 Analysis

We assess the impact of proposed components and various influencing factors, as detailed in Table 4.

Impact of proposed components. Our ablation study confirms the significance of each component in our approach. Removing the semi-parametric loss (w/o SP objective) leads to a drop in accuracy of 5.3% for SIDR$_{\text{full}}$ and a substantial 31.5% for SIDR$_\beta$, rendering beta search non-functional. Moreover, excluding in-training retrieved negatives (w/o retrieved neg) results in a decrease in top-1 accuracy: 4.2% for SIDR$_{\text{full}}$ and 7.5% for SIDR$_\beta$. These results highlight the effectiveness of using beta search for in-training retrieval. Unlike static BM25 negatives, which quickly diminish in effectiveness as the model learns, beta search utilizes parametric queries that evolve with the model, continually ensuring that the negatives are challenging and relevant throughout the training process.

Table 4: Ablation study of SIDR$_{\text{full}}$ and SIDR$_\beta$ on NQ dataset.

	top1	top5	top20
Parametric Index			
SIDR$_{\text{full}}$	**49.1**	**69.3**	**80.7**
w/ retrieved neg (m=1)	47.9	68.4	79.6
w/ retrieved neg (m=100)	47.3	69.0	80.5
w/ retrieved neg (MARCO)	39.7	63.9	77.5
w/ retrieved neg (WIKI 8m)	48.3	69.1	80.6
w/o retrieved neg	44.9	66.9	78.8
w/o neg	30.2	57.4	75.1
w/o SP objective	43.8	68.0	79.9
Non-parametric Index			
SIDR$_\beta$	39.8	**62.9**	**76.3**
w/ retrieved neg (m=1)	**41.2**	62.3	76.4
w/ retrieved neg (m=100)	37.3	62.4	76.5
w/ retrieved neg (MARCO)	29.5	54.8	70.1
w/ retrieved neg (WIKI 8m)	37.5	62.4	76.3
w/o retrieved neg	32.3	56.0	72.1
w/o neg	24.4	49.1	68.2
w/o SP objective	12.3	30.0	46.8
w/ vary lentgh	37.5	61.2	76.1

Effect of negative sample hardness. We explore how the difficulty of retrieved negatives affects model effectiveness. The parameter m indicates the size of the passage pool from which negatives are identified and then randomly drawn, with lower m values yielding harder negatives. While these harder negatives can improve contrastive learning, they also increase the risk of misclassifying weak positives as negatives. Our results (w/ retrieved neg m={1,100}) indicate that adjusting m to 1 or 100, compared to the baseline of 20, degrades the performance. Thus, an m value of 20 provides the optimal balance, effectively challenging the model while reducing the likelihood of misclassification.

Effect of negative sample source. Our results (w/ retrieved neg MARCO) indicate that switching the source of negative samples from the Wikipedia corpus to the MS MARCO with 8.8 million passages leads to a notable drop in performance. In a parallel experiment (w/ retrieved neg WIKI 8m) using a Wikipedia corpus of the same size, performance remained consistent with our baseline, indicating that corpus size is not the main factor behind the observed decline. Instead, the source of negatives plays a crucial role in performance. The disparity stems from differences in the writing styles and structures unique to each corpus (e.g., Wikipedia passages typically include a short title preceding the text), which cause the model to focus on superficial, corpus-specific features rather than developing a deeper understanding of the relevance.

Impact of text length on non-parametric index. Unlike the BM25 term-based index, the BoT index lacks term weighting, meaning longer texts may activate more dimensions, resulting in higher inner product scores. To assess the impact of text length on the effectiveness of SIDR$_\beta$, we re-segmented the Wikipedia corpus into passages ranging from 50 to 200 words, while maintaining the same overall number of passages. Our results (w/ vary lentgh) show that the top-1 accuracy of SIDR$_\beta$ decreased slightly from 39.8% to 37.5%, indicating minimal impact on performance. This slight drop can be explained by the sub-linear growth in unique tokens as text length increases and the high sparsity of the representations, where increasing activations has little impact on relevance.

Comparison of in-training retrieval. We compared in-training retrieval across different systems, as shown in Table 5. Compared to BM25, SIDR$_\beta$ has up to 30x lower latency when using GPU resources for large corpora. In contrast to dense retrieval methods like DPR, SIDR$_\beta$ requires less GPU allocation and uses a fixed index, which eliminates the need for periodic re-indexing during the training loop and ensures that the training objective is not compromised by a stale index. Additional details can be found in Appendix C.

Table 5: In-training retrieval latency per batch, with the storage size and GPU memory allocation for corresponding index.

Method	Latency	Storage	GPU
BM25	3s	2.3GB	/
DPR	<1ms	31.5GB	31GB
SIDR$_\beta$	<1ms	2.7GB	10GB

7 Related Work

Disentangled Retrieval Disentangled retrieval, also known as sparse lexical retrieval, develops sparse representations for queries and documents within a pre-defined vocabulary space, where each dimension reflecting the importance of a specific token. These methods have proven effective in text matching (Dai & Callan, 2020; Bai et al., 2020; Formal et al., 2021b; 2022; Ram et al., 2022) and have been utilized to enhance search efficiency in subsequent studies (Gao et al., 2021a; Shen et al., 2022; Lin et al., 2023b; Lin & Lin, 2023). Notably, several works like TILDE (Zhuang & Zuccon, 2021b;a), and SPARTA (Zhao et al., 2021) use bag-of-tokens query representations for efficient online query processing. These methods fall under the category of semi-parametric retrieval as they employ non-parametric representations on the query side. Complementing these efforts, our work focuses on addressing the challenges associated with the index side, which is inherently more complex due to the greater length and contextual depth of documents. We discuss the taxonomy of neural retrieval in more detail in Appendix A.

In-training Retrieval Retrieving data in the training loop of retrieval models is an emerging yet challenging practice that serves several critical purposes. This includes acquiring negative samples for contrastive learning (Zhan et al., 2021; Robinson et al., 2021), sourcing relevant instances for data augmentation (Blattmann et al., 2022; Shi et al., 2023), and facilitating the training of retrieval-based language models (Asai et al., 2023) in an end-to-end manner. However, this process is complicated due to the need for frequent re-indexing of the corpus as the training of the retriever progresses. Recent research has explored strategies like asynchronous index updates (Guu et al., 2020; Xiong et al., 2020; Izacard et al., 2022b; Shi et al., 2023) or building temporary indexes on-the-fly from the current training batch (Zhong et al., 2022; Min et al., 2022). Our work proposes a semi-parametric framework which supports a non-parametric index, thereby avoiding these complications and streamlining the in-training retrieval practice.

8 Conclusions

In this paper, we introduce SIDR, a semi-parametric bi-encoder retrieval framework that supports both parametric and non-parametric indexes to address the emerging needs of retrieval-based applications. Unlike traditional neural retrieval methods that rely solely on embeddings as indexes, SIDR additionally incorporates a non-parametric bag-of-tokens index. The flexibility of SIDR makes it particularly well-suited for applications requiring efficient or low-cost indexing and facilitates co-training with a fixed index.

Acknowledgments

Lei Chen's work is partially supported by National Key Research and Development Program of China Grant No. 2023YFF0725100, National Science Foundation of China (NSFC) under Grant No. U22B2060, Guangdong-Hong Kong Technology Innovation Joint Funding Scheme Project No. 2024A0505040012, the Hong Kong RGC GRF Project 16213620, RIF Project R6020-19, AOE Project AoE/E-603/18, Theme-based project TRS T41-603/20R, CRF Project C2004-21G, Guangdong Province Science and Technology Plan Project 2023A0505030011, Guangzhou municipality big data intelligence key lab, 2023A03J0012, Hong Kong ITC ITF grants MHX/078/21 and PRP/004/22FX, Zhujiang scholar program 2021JC02X170, Microsoft Research Asia Collaborative Research Grant, HKUST-Webank joint research lab and 2023 HKUST Shenzhen-Hong Kong Collaborative Innovation Institute Green Sustainability Special Fund, from Shui On Xintiandi and the InnoSpace GBA.

References

Simran Arora, Patrick Lewis, Angela Fan, Jacob Kahn, and Christopher Ré. Reasoning over public and private data in retrieval-based systems. *Transactions of the Association for Computational Linguistics*, 11:902–921, 2023.

Akari Asai, Sewon Min, Zexuan Zhong, and Danqi Chen. Retrieval-based language models and applications. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 6: Tutorial Abstracts)*, pp. 41–46, 2023.

Yang Bai, Xiaoguang Li, Gang Wang, Chaoliang Zhang, Lifeng Shang, Jun Xu, Zhaowei Wang, Fangshan Wang, and Qun Liu. Sparterm: Learning term-based sparse representation for fast text retrieval. *arXiv preprint arXiv:2010.00768*, 2020.

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

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1533–1544, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://aclanthology.org/D13-1160.

Andreas Blattmann, Robin Rombach, Kaan Oktay, Jonas Müller, and Björn Ommer. Retrieval-augmented diffusion models. *Advances in Neural Information Processing Systems*, 35:15309–15324, 2022.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*, 2017.

Zhuyun Dai and Jamie Callan. Context-aware term weighting for first stage passage retrieval. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pp. 1533–1536, 2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Manqing Dong, Feng Yuan, Lina Yao, Xiwei Xu, and Liming Zhu. Mamo: Memory-augmented meta-optimization for cold-start recommendation. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 688–697, 2020.

Yixing Fan, Xiaohui Xie, Yinqiong Cai, Jia Chen, Xinyu Ma, Xiangsheng Li, Ruqing Zhang, Jiafeng Guo, et al. Pre-training methods in information retrieval. *Foundations and Trends® in Information Retrieval*, 16(3):178–317, 2022.

Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. Splade v2: Sparse lexical and expansion model for information retrieval. *arXiv preprint arXiv:2109.10086*, 2021a.

Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 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*, pp. 2288–2292, 2021b.

Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. 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*, pp. 2353–2359, 2022.

William T Freeman, Thouis R Jones, and Egon C Pasztor. Example-based super-resolution. *IEEE Computer graphics and Applications*, 22(2):56–65, 2002.

Luyu Gao, Zhuyun Dai, and Jamie Callan. Coil: Revisit exact lexical match in information retrieval with contextualized inverted list. *arXiv preprint arXiv:2104.07186*, 2021a.

Luyu Gao, Zhuyun Dai, Tongfei Chen, Zhen Fan, Benjamin Van Durme, and Jamie Callan. Complement lexical retrieval model with semantic residual embeddings. In *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28–April 1, 2021, Proceedings, Part I 43*, pp. 146–160. Springer, 2021b.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In *International conference on machine learning*, pp. 3929–3938. PMLR, 2020.

Jie Huang, Hanyin Shao, and Kevin Chen-Chuan Chang. Are large pre-trained language models leaking your personal information? *arXiv preprint arXiv:2205.12628*, 2022.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*, 2021.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: Few-shot learning with retrieval augmented language models. *arXiv preprint arXiv:2208.03299*, 2022a.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Few-shot learning with retrieval augmented language models. *arXiv preprint arXiv:2208.03299*, 2022b.

Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2020.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1147. URL https://aclanthology.org/P17-1147.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.550. URL https://aclanthology.org/2020.emnlp-main.550.

Oleksandr Kolomiyets and Marie-Francine Moens. A survey on question answering technology from an information retrieval perspective. *Information Sciences*, 181(24):5412–5434, 2011.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

Jurek Leonhardt, Koustav Rudra, Megha Khosla, Abhijit Anand, and Avishek Anand. Efficient neural ranking using forward indexes. In *Proceedings of the ACM Web Conference 2022*, pp. 266–276, 2022.

Jimmy Lin and Xueguang Ma. A few brief notes on deepimpact, coil, and a conceptual framework for information retrieval techniques. *arXiv preprint arXiv:2106.14807*, 2021.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. Pyserini: A python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2356–2362, 2021.

Sheng-Chieh Lin and Jimmy Lin. A dense representation framework for lexical and semantic matching. *ACM Transactions on Information Systems*, 41(4):1–29, 2023.

Sheng-Chieh Lin, Akari Asai, Minghan Li, Barlas Oguz, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, and Xilun Chen. How to train your dragon: Diverse augmentation towards generalizable dense retrieval. *arXiv preprint arXiv:2302.07452*, 2023a.

Sheng-Chieh Lin, Minghan Li, and Jimmy Lin. Aggretriever: A simple approach to aggregate textual representations for robust dense passage retrieval. *Transactions of the Association for Computational Linguistics*, 11:436–452, 2023b.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024a.

Xiao-Yang Liu, Guoxuan Wang, and Daochen Zha. Fingpt: Democratizing internet-scale data for financial large language models. *arXiv preprint arXiv:2307.10485*, 2023.

Zihan Liu, Wei Ping, Rajarshi Roy, Peng Xu, Chankyu Lee, Mohammad Shoeybi, and Bryan Catanzaro. Chatqa: Surpassing gpt-4 on conversational qa and rag. *arXiv preprint arXiv:2401.10225*, 2024b.

Alexander Long, Wei Yin, Thalaiyasingam Ajanthan, Vu Nguyen, Pulak Purkait, Ravi Garg, Alan Blair, Chunhua Shen, and Anton van den Hengel. Retrieval augmented classification for long-tail visual recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6959–6969, 2022.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. Fine-tuning llama for multi-stage text retrieval. *arXiv preprint arXiv:2310.08319*, 2023.

Christopher D Manning. *An introduction to information retrieval*. Cambridge university press, 2009.

Ahtsham Manzoor and Dietmar Jannach. Towards retrieval-based conversational recommendation. *Information Systems*, 109:102083, 2022.

Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ram Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, et al. Augmented language models: a survey. *arXiv preprint arXiv:2302.07842*, 2023.

Sewon Min, Weijia Shi, Mike Lewis, Xilun Chen, Wen-tau Yih, Hannaneh Hajishirzi, and Luke Zettlemoyer. Nonparametric masked language modeling. *arXiv preprint arXiv:2212.01349*, 2022.

Sewon Min, Suchin Gururangan, Eric Wallace, Hannaneh Hajishirzi, Noah A Smith, and Luke Zettlemoyer. Silo language models: Isolating legal risk in a nonparametric datastore. *arXiv preprint arXiv:2308.04430*, 2023.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernández Ábrego, Ji Ma, Vincent Y Zhao, Yi Luan, Keith B Hall, Ming-Wei Chang, et al. Large dual encoders are generalizable retrievers. *arXiv preprint arXiv:2112.07899*, 2021.

Ori Ram, Liat Bezalel, Adi Zicher, Yonatan Belinkov, Jonathan Berant, and Amir Globerson. What are you token about? dense retrieval as distributions over the vocabulary. *arXiv preprint arXiv:2212.10380*, 2022.

Juan Ramos et al. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pp. 29–48. Citeseer, 2003.

Stephen Robertson, Hugo Zaragoza, et al. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009.

Joshua David Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with hard negative samples. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=CR1XOQ0UTh-.

Rulin Shao, Jacqueline He, Akari Asai, Weijia Shi, Tim Dettmers, Sewon Min, Luke Zettlemoyer, and Pang Wei Koh. Scaling retrieval-based language models with a trillion-token datastore. *arXiv preprint arXiv:2407.12854*, 2024.

Tao Shen, Xiubo Geng, Chongyang Tao, Can Xu, Xiaolong Huang, Binxing Jiao, Linjun Yang, and Daxin Jiang. Lexmae: Lexicon-bottlenecked pretraining for large-scale retrieval. *arXiv preprint arXiv:2208.14754*, 2022.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. Replug: Retrieval-augmented black-box language models. *arXiv preprint arXiv:2301.12652*, 2023.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. Retrieval augmentation reduces hallucination in conversation. *arXiv preprint arXiv:2104.07567*, 2021.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 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)*, 2021.

Cunxiang Wang, Xiaoze Liu, Yuanhao Yue, Xiangru Tang, Tianhang Zhang, Cheng Jiayang, Yunzhi Yao, Wenyang Gao, Xuming Hu, Zehan Qi, et al. Survey on factuality in large language models: Knowledge, retrieval and domain-specificity. *arXiv preprint arXiv:2310.07521*, 2023.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):1–26, 2024.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *arXiv preprint arXiv:2007.00808*, 2020.

Ikuya Yamada, Akari Asai, and Hannaneh Hajishirzi. Efficient passage retrieval with hashing for open-domain question answering. *arXiv preprint arXiv:2106.00882*, 2021.

Wenhao Yu, Chenguang Zhu, Zaitang Li, Zhiting Hu, Qingyun Wang, Heng Ji, and Meng Jiang. A survey of knowledge-enhanced text generation. *ACM Computing Surveys*, 54(11s):1–38, 2022.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. Optimizing dense retrieval model training with hard negatives. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1503–1512, 2021.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. Siren's song in the ai ocean: A survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*, 2023.

Tiancheng Zhao, Xiaopeng Lu, and Kyusong Lee. Sparta: Efficient open-domain question answering via sparse transformer matching retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 565–575, 2021.

Zexuan Zhong, Tao Lei, and Danqi Chen. Training language models with memory augmentation. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.382. URL https://aclanthology.org/2022.emnlp-main.382.

Jiawei Zhou, Xiaoguang Li, Lifeng Shang, Lan Luo, Ke Zhan, Enrui Hu, Xinyu Zhang, Hao Jiang, Zhao Cao, Fan Yu, et al. Hyperlink-induced pre-training for passage retrieval in open-domain question answering. *arXiv preprint arXiv:2203.06942*, 2022.

Jiawei Zhou, Xiaoguang Li, Lifeng Shang, Xin Jiang, Qun Liu, and Lei Chen. Retrieval-based disentangled representation learning with natural language supervision. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=ZlQRiFmq7Y.

Fengbin Zhu, Wenqiang Lei, Chao Wang, Jianming Zheng, Soujanya Poria, and Tat-Seng Chua. Retrieving and reading: A comprehensive survey on open-domain question answering. *arXiv preprint arXiv:2101.00774*, 2021.

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

Shengyao Zhuang and Guido Zuccon. Fast passage re-ranking with contextualized exact term matching and efficient passage expansion. *arXiv preprint arXiv:2108.08513*, 2021a.

Shengyao Zhuang and Guido Zuccon. Tilde: Term independent likelihood model for passage re-ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1483–1492, 2021b.

A Taxonomy of Neural Retrieval

In this section, we outline the taxonomy of existing neural retrievers, discussing distinctions such as disentangled versus entangled, dense versus sparse, and parametric versus semi-parametric. This classification aims to clarify the concepts discussed throughout our paper.

Entangled vs. Disentangled Retriever differ in the type of representations they use for search:

- **Entangled retriever** typically employs latent representations with dimensions such as 512, 768, 1024, or 2048 for inner-product search. These representations can be learned flexibly and effectively without prior knowledge, as they are entangled. An entangled retriever can also use sparse latent representations, such as the BPR method (Yamada et al., 2021), which employs a learned hash function on a 768-dimensional latent vector for efficient searching.
- **Disentangled retriever**, on the other hand, utilizes disentangled representations with much larger dimensionality, spanning from tens of thousands, representing the language model's vocabulary, to millions, typical of BM25 vocabulary. Each dimension corresponds to the importance of a specific token within the vocabulary. Disentangled retrieval methods rely on disentangled representations as a prior, which leads to desirable properties such as interpretability, controllability, robustness, and, as discussed in this paper, accelerated indexing. Previous work (Zhou et al., 2024; Formal et al., 2021a; Shen et al., 2022) has also suggested that using disentangled representations improves embedding quality, as foundational models are optimized in this disentangled representation space to predict mask tokens or next tokens, rather than in the entangled representation space. Note

that disentangled retrievers can also use dense (fully activated) representations. For example, VDR (Zhou et al., 2024) utilizes disentangled representations that can be fully activated for search.

Thus, the classification of entangled vs. disentangled is independent of whether the representation is dense or sparse. Generally, entangled retrieval methods tend to use dense representations, while disentangled retrieval methods often use sparse ones, but this is not a strict requirement.

Dense vs. Sparse Retriever differ based on whether their representations are fully activated or have been sparsified. Typically, entangled representations are fully activated, while disentangled representations are often sparsified to reduce storage size and to facilitate the construction of an inverted index for efficient searching. Consequently, dense retrieval is commonly associated with entangled representations and sparse retrieval with disentangled ones. However, the relationship between entangled/disentangled and dense/sparse retrieval is not rigid. Exceptions exist where entangled representations can be sparsified (Yamada et al., 2021) and disentangled representations can be fully activated (Zhou et al., 2024).

Full Parametric vs. Semi-parametric Retrieval These approaches differ based on the utilization of neural parameters for encoders. Full parametric retrieval systems employ neural parameters for both encoders. In contrast, semi-parametric retrieval systems use one neural encoder alongside one non-parametric encoder, typically involving tokenization-based representations. To our knowledge, existing semi-parametric systems predominantly engage in disentangled retrieval, as they all utilize tokenization-based non-parametric representations that operate on a vocabulary space. Notable examples include TILDE (Zhuang & Zuccon, 2021b), TILDE-v2 (Zhuang & Zuccon, 2021a), and SPARTA (Zhao et al., 2021), which implement tokenization-based representations on the query side for efficient online query processing. Our work, SIDR, represents a unique approach within this category, offering semi-parametric retrieval that utilizes binary bag-of-tokens representations on the index side for emerging scenarios.

B Details of Latency Evaluation

During the indexing stage, E_θ and T operate with a batch size of 32 and a maximum text length of 256. In the search stage, the computation of $\mathrm{f}(q, \mathcal{D})$ includes both inner product calculation and sorting of the top-k passages. Queries are processed in batches of 32, with passage embeddings stored in either CPU or GPU memory using half-precision floating-point (FP16) to optimize memory usage. Our analysis excludes the time spent on I/O and data type conversion between CPU and GPU, assuming sufficient processing resources are available.

For BM25, we utilize Pyserini (Lin et al., 2021), a library based on a Java implementation developed around Lucene. For neural retrieval, our implementation is in Python, leveraging PyTorch's sparse module[1] for efficient inner product computation, without building an inverted index. Timing measurements are performed by running each operation 10 times and reporting the average after excluding the maximum and minimum values. To avoid out-of-memory issues on our devices, we perform searches in batches on a 1-million document corpus and accumulate the latencies. Note that inverted indexes rely heavily on memory access and integer operations, which are generally inefficient on GPU architectures.

C In-training Retrieval Details

We conducted a simulation test to evaluate factors affecting in-training retrieval, summarized in Table 6. Initially, we built a binary token index with dimensions of 30k and a sample size of 500m, where each vector consists of 256 dimensional activated. We then varied the density of passage representations by adjusting the activation number from 256 to 512 and 1024. As the activation number increased, storage and GPU alloca-

	Latency (ms)	Storage (GB)	GPU (GB)
	Index Density		
a=256	0.20	2.8	6.9
a=512	0.21	4.9	23.5
a=1024	0.21	9.0	46.7
	Query Batch Size		
bs=32	0.20	/	/
bs=128	0.21	/	/
bs=512	0.24	/	/

Table 6: Retrieval latency, index storage size, and GPU allocation for SIDR$_\beta$.

[1] https://pytorch.org/docs/stable/sparse.html

tion also increased, while latency remained largely unchanged. Additionally, we found query batch size had minimal impact on latency.

D Analysis on Term Weighting and Expansion

To systematically compare our method with BM25, we control for vocabulary differences by using BM25 with the same BERT-base-uncased vocabulary as ours. This ensures both methods share the same dimensionality for sparse representation, differing only in two key aspects:

- Term expansion: BM25 uses only lexical tokens, while SiDR allows for term expansion.
- Term weighting: BM25 relies on statistical-based term weights, whereas SiDR learns contextualized weights.

In this section, we empirically demonstrate how these factors – term expansion and term weighting – affect the outcome. Below, we introduce two additional representation forms:

Lexical Parametric Representation $V_\theta^{lex}(x)$ This representation activates only the lexical tokens present in the input text x. It leverages learned term weights but does not permit term expansion:

$$V_\theta^{lex}(x) = V_{\text{BoT}} \circ V_\theta(x)$$

Binary Parametric Representation $V_\theta^{bin}(x)$ This representation activates the same number of tokens but assigns uniform weights (set to one), removing learned scalar term weighting. It allows term expansion but does not apply term weights:

$$V_\theta^{bin}(x) = \text{Binarize} \circ V_\theta(x)$$

where Binarize is a function mapping non-zero values to one.

To independently assess the impact of term expansion and term weighting, we propose several variants of SiDR$_{\text{full}}$: SiDR$_{\text{full}}$ (w/o weight at doc) and SiDR$_{\text{full}}$ (w/o expand at doc), which utilize $V_\theta^{bin}(p)$ and $V_\theta^{lex}(p)$ on the document side during inference. Additionally, we introduce variants of SiDR$_\beta$: SiDR$_\beta$ (w/o weight at query) and SiDR$_\beta$ (w/o expand at query), which employ $V_\theta^{bin}(q)$ and $V_\theta^{lex}(q)$ on the query side at inference. Furthermore, we propose SiDR$_{full}$ (w/o expand at doc, training), which is trained with $V_\theta(p)$ replaced by $V_\theta^{\text{lex}}(p)$ to ensure consistency between training and inference phases. All these models are compared against BM25 that utilizes the same bert-base-uncased tokenization and vocabulary. This controls for vocabulary differences, isolating the effects of term selection and term weighting. We also include an extreme baseline that uses a bag-of-tokens representation for both the query and the passage, referred to as BoT overlap.

Table 7: Ablation study on learned term weighting and term expansion, with results reported as top-1 accuracy on NQ test splits.

Model	Query		Document		Accuracy
	Expand	Weight	Expand	Weight	
BM25 (bert-base-uncased)	×	✓	×	✓	21.9
Ablation of SiDR$_{full}$ on doc side					
SiDR$_{\text{full}}$	✓	✓	✓	✓	49.1
SiDR$_{\text{full}}$ (w/o weight at doc)			✓	×	33.1
SiDR$_{\text{full}}$ (w/o expand at doc)			×	✓	38.9
SiDR$_{\text{full}}$ (w/o expand at doc, training)			×	✓	43.1
SiDR$_{\text{beta}}$			×	×	39.8
Ablation of SiDR$_{beta}$ on query side					
SiDR$_{\text{beta}}$ (w/o weight at query)	✓	×	×	×	14.5
SiDR$_{\text{beta}}$ (w/o expand at query)	×	✓			34.3
BoT overlap	×	×			14.2

When using a parametric index, we conduct an ablation study on SiDR$_{\text{full}}$ to assess the impact of removing term weighting and term expansion on the document side. From top to bottom, we systematically remove term weight or expansion, simplifying the index of SiDR$_{\text{full}}$ to assess their individual contributions. Our results indicate that removing either term weight or term expansion leads to worse outcomes than removing both (i.e., SiDR$_\beta$). This is because our training objective is specifically designed to align query embeddings with the BoT index, rather than these variations. Furthermore, we find that if the training is adjusted to accommodate these variations, such as document representations without term expansion, these variations can outperform the BoT index. This demonstrates that neural bi-encoders have great learning potential, with improvement stemming from not only the training itself but also how well the training aligns with inference.

When using a bag-of-tokens index, we assess the impact of term weight and expansion on the query side. Starting from a baseline uses unweighted term overlap, referred to as "BoT overlap", applying BM25's term weights to both queries and documents yields a 7.7% improvement. In comparison, our method's learned query term weights achieve a 20.1% improvement, while learned term expansion provides minimal additional gain. Combining term weights and expansion on the query side results in a 25.6% improvement, which is SiDR$_\beta$.

In conclusion, when using an embedding index, we demonstrate that both learned term weighting and term expansion on the document side are crucial. Conversely, when using a bag-of-tokens index, the improvements primarily come from term weighting on the query side rather than expansion. Furthermore, ensuring consistency between training and inference representations is essential. A parametric index can underperform compared to non-parametric ones if the representations used during inference do not align with those used during training.

E Cost-Effectiveness Analysis

Table 8: Additional late-parametric baselines and cost-effectiveness analysis performed on a retrieval task using 3.6k NQ test queries across a 21 million Wikipedia corpus. Costs are determined by the number of text chunks (including both queries and passages) that require embedding by neural encoders. Parentheses indicate the ratio of text chunks embedded to the total retrieval corpus.

	Performance	Cost
Non-parametric Index		
BM25	22.7	0
SiDR$_\beta$	39.8	3.6k (0.01%)
Late-parametric with top-100 rerank		
BM25 + VDR	41.6	364k (1.73%)
BM25 + SiDR	44.0	364k (1.73%)
BM25 + Contriever	39.3	364k (1.73%)
BM25 + E5$_{\text{base}}$	50.4	364k (1.73%)
SiDR$_\beta$ (m=100)	50.3	364k (1.73%)
SiDR$_\beta$ + VDR	43.2	367k (1.74%)
SiDR$_\beta$ + Contriever	42.9	367k (1.74%)
SiDR$_\beta$ + E5$_{\text{base}}$	57.7	367k (1.74%)
Parametric Index		
SiDR$_{\text{full}}$	49.1	21m (100.01%)
Contriever	41.5	21m (100.01%)
E5$_{\text{base}}$	57.9	21m (100.01%)

Late parametric retrieval aims to provide a quick-start and low-cost search initialization through a non-parametric index, while simultaneously building a parametric index during the search service, eventually transitioning to a fully parametric index for searching. To fulfill this requirement, the first-stage utilizes a retriever that supports a non-parametric index, while the second-stage retriever can be any parametric bi-encoder. This method can be seen as a subset of hybrid retrieval systems (Leonhardt et al., 2022; Gao et al., 2021b), with specific choices constrained to the two stages.

We introduce various combinations of BM25 and SiDR$_\beta$ as the first-stage retriever, paired with more advanced retrievers in the second stage to demonstrate their effectiveness. The results of these combinations are presented in Table 8. Moreover, we assess the cost-effectiveness of these frameworks, particularly in scenarios where raw data has not been indexed. In such cases, the primary cost arises from the neural model's forward pass for text embedding. Therefore, we measure cost by counting the number of text chunks (both query and passage) that require embedding.

For BM25, no neural embedding is required. While SiDR$_\beta$ employs a bag-of-tokens index, waiving the indexing cost, it requires embedding 3.6k queries (0.01% of the corpus) to complete the retrieval task. Despite this, it offers a significant performance improvement of 17.1% in accuracy over BM25. For various late parametric baselines, an additional embedding of 100 passages per query is needed — approximately 1.7% of the corpus — yet this results in further improvements over BM25. Conversely, the conventional retrieval pipeline, which requires embedding the entire corpus to achieve performance comparable to that of late parametric models with top-100 reranking. This analysis shows that semi-parametric models provide a more cost-effective solution by balancing retrieval performance with computational efficiency.

Among various late-parametric baselines, SiDR$_\beta$ consistently outperforms BM25 as the first-stage retriever. However, this advantage requires embedding an additional 3.6k queries if other models are employed as the second-stage retriever. In exploring second-stage retrievers, we have tested state-of-the-art models like E5 and Contriever. Our results indicate that stronger retrievers lead to better overall late-parametric performance. Notably, in all our tests, SiDR$_\beta$ combined with any second-stage model consistently outperforms BM25 paired with the same model. Furthermore, when SiDR$_\beta$ serves as the first-stage retriever and re-ranks the top-100 passages, its performance is comparable to, and often exceeds, that of full parametric search with these retrievers.