COILcr: Efficient Semantic Matching in Contextualized Exact Match Retrieval

Zhen Fan, Luyu Gao, Rohan Jha, and Jamie Callan

Carnegie Mellon University, Pittsburgh PA 15213, USA

Abstract. Lexical exact match systems that use inverted lists are a fundamental text retrieval architecture. A recent advance in neural IR, COIL, extends this approach with *contextualized inverted lists* from a deep language model backbone and performs retrieval by comparing contextualized query-document term representation, which is effective but computationally expensive. This paper explores the effectiveness-efficiency tradeoff in COIL-style systems, aiming to reduce the computational complexity of retrieval while preserving term semantics. It proposes COILcr, which explicitly factorizes COIL into intra-context term importance weights and cross-context semantic representations. At indexing time, COILcr further maps term semantic representations to a smaller set of canonical representations. Experiments demonstrate that canonical representations can efficiently preserve term semantics, reducing the storage and computational cost of COIL-based retrieval while maintaining model performance. The paper also discusses and compares multiple heuristics for canonical representation selection and looks into its performance in different retrieval settings.

Keywords: first-stage retrieval, lexical exact match, deep language models, contextualized inverted lists, approximation

1 Introduction

Lexical exact matching [21] has been a fundamental component of classic information retrieval (IR). In the new era of neural IR and deep language models (LM) [24,2], lexical retrievers are still used in large-scale settings and initial stages of ranking pipelines due to their simplicity and efficiency: lexical exact match signals are captured at the token (word) level, and the matching process can be highly accelerated with *inverted indexes* built during offline preprocessing. Such simplicity, however, is accompanied by the natural gap between explicit lexical form and the implicit semantics of a concept. Lexical retrievers have long suffered from the *vocabulary mismatch* (different lexical forms for the same concept) and *semantic mismatch* (different semantic meanings for the same lexical form) between the two spaces.

The introduction of deep LMs has led to large improvements in search accuracy [18]. Deep LM-augmented lexical retrieval systems fine-tune pretrained language models on different retrieval-specific tasks to aid classic non-neural systems or directly perform retrieval. Particularly, to address semantic mismatch, recent work uses deep LMs to generate contextualized term representations and calculates term scores by vector similarity instead of scalar weight product.

Gao et al. proposed COIL (contextualized inverted lists)[8], a framework for incorporating semantic matching in lexical exact match systems. COIL augments traditional inverted lists with contextualized representations of each term occurrence. At search time, it keeps the constraint of lexical exact match, but calculates term scores based on the similarity of the representations. Compared to previous systems using term weights to model *in-sequence term importance*, COIL uses term representation vectors to additionally measure *cross-sequence term similarity* between query and document terms. This leads to improved retrieval accuracy but also increased storage and computational costs. Every term is represented by a d-dimensional vector during indexing, and each term score is calculated by a d-dimensional vector dot product at retrieval time.

This paper builds on the COIL framework, focusing on its semantic matching capability. It investigates whether it is it possible to effectively recognize semantic mismatch at a lower cost. COIL utilizes a dense vector to directly model a term's importance as well as the fine-grained semantics of its unique context. However, for a vocabulary term, the number of its important meanings or base senses across the corpus is usually much smaller than its actual collection term frequency. While modeling fine-grained term semantic match requires precise vector similarity comparison, modeling the mismatch of coarse term senses may not require such high representation capacity, and can be performed more efficiently by approximation of term representations.

Following these ideas, we propose COILcr, **CO**ntextualized **I**nverted **L**ists with **C**anonical **R**epresentations, to efficiently model coarse term semantics in COIL-based lexical exact match systems. We first factorize term representations and decouple term importance weight and semantic representation. We proceed to build a set of canonical representations (CR) for term semantic representations via spherical k-means clustering [3], and map all individual term occurrences to the set of base CRs. This approximation reduces the inverted index storage size and number of similarity calculations at retrieval time. We demonstrate through multiple experiments that COILcr's approximation is almost as effective as precise lexical COIL systems, but at much lower storage and computational cost.

The next section discusses related work, and provides a detailed description of the COIL framework. Section 3 describes the proposed canonical representation-based approach to the recognition of semantic mismatch. Section 4 discusses our experimental methodology, and Section 5 discusses experiment results and findings. Section 6 concludes.

2 Related Work

The introduction of deep language models [2] has led to a new era in neural IR. Cross-encoder models [18] first demonstrated that deep LMs can be tuned to understand context and handle the gap between explicit text and implicit semantics, and achieve state-of-the-art performance in neural reranking tasks.

Large-scale neural ranking systems have also benefited from the finetuned LMs' capability to generate semantic representations both at the text sequence level and at the lexical token level. *Dense retrieval* systems [14] directly encode text segments into a dense semantic representation, and score query-document pairs with some vector similarity metric.

$$\mathcal{S}(q,d) = \sigma(\mathbf{v}_q, \mathbf{v}_d)$$

where $\mathbf{v}_q$ and $\mathbf{v}_d$ are dense representations of the query and document, usually the [CLS] outout of the language model, and σ is a similarity function such as dot product or cosine similarity. Recent work investigates various training techniques to improve the quality of representations, including hard negative mining [11,25], pretraining task design [6,7,11] and knowledge distillation [9,10,20].

Lexical match systems, on the other hand, perform encoding and matching at the lexical token level, and score query-document pairs by aggregating term match scores.

$$\mathcal{S}(q,d) = \sum_{t \in \mathcal{V}_q \cap \mathcal{V}_d} s_t(q,d)$$

where $\mathcal{S}(q,d)$ is the overall document score, $s_t(q,d)$ is term matching score of term t, and $\mathcal{V}_q$ and $\mathcal{V}_d$ are the sets of terms in the query and document respectively. For non-neural lexical exact match retrievers such as BM25 [21], a document term is represented by a scalar weight $w_{t,d}$ that represents its *importance* and is stored in an inverted list. Term scoring is modeled as $s_t(q,d) = w_{t,q} w_{t,d}$, a product of query and document term importance weights. Finetuned language models were first introduced to improve existing non-neural weighting-based systems by performing term reweighting [1] and explicit vocabulary expansion [19].

In such lexical exact match systems, storing scalar weights ensures efficient storage and computational cost, but does not preserve extra semantic information or context. At retrieval time, the system can not distinguish the actual semantic agreement between query and document terms, thereby suffering from semantic mismatch. To tackle this problem, researchers explored using contextualized representations in lexical retrieval [29,15,22] under soft match settings. Gao et al. further proposed COIL[1] [8], which introduces contextualized term representations under lexical exact match settings, and expands the term weight product into a vector similarity calculation to further model the semantic simi-

[1] The full COIL retrieval model is a hybrid model combining dense document scoring and sparse token scoring. In this paper we mainly focus on the lexical exact match retrieval setting, and mainly refer to COIL as the basic concept of contextualized term representation and inverted index. We compare our system to the lexical-only model form of the COIL retriever, referred to as COIL-*tok* in the original work.

larity between query and document terms.

$$\begin{aligned} \mathbf{v}_{q_i} &= \phi(LM(q,i)) \\ \mathbf{v}_{d_j} &= \phi(LM(d,j)) \\ s_t(q_i,d_j) &= \mathbf{v}_{q_i}^{\intercal}\mathbf{v}_{d_j} \end{aligned}$$

where $q_i = d_j$ are the i-th query term and j-th document term with vector representation $\mathbf{v}_{q_i}$ and $\mathbf{v}_{d_j}$ respectively. ϕ denotes a linear transformation layer that maps the LM output to token representations of a lower dimension.

COIL's lexical-only model variant COIL-tok is a natural extension of traditional lexical retrieval systems. The vector representations of document terms $\mathbf{v}_{d_j}$ are precomputed and indexed in inverted lists, and the overall query-document score is the aggregation of exact match term scores. Replacing term weights with vectors leads to clear performance gain in accuracy and recall but also increases storage and computational cost. Lin and Ma further proposed UniCOIL [16], a followup study to COIL in which the generated representation dimension is lowered to $d_v = 1$ and the COIL language model directly predicts scalar term weight, and demonstrates that the model achieves decent accuracy with much lower cost under term weighting-only settings.

In this paper, we look into the necessity and methodology of preserving term semantics in COIL-style systems and balancing its effectiveness-efficiency tradeoff. Index compression and retrieval efficiency has gained growing research interest with the development of neural IR systems. Recent systems investigate multiple methods such as dimension reduction [12], hashing [26], product quantization [28,27] and residual compression [22].

3 COIL$_{\text{CR}}$: Contextualized Inverted Lists with Canonical Representations

COILcr is based on two key ideas: i) factorizing COIL token representations into intra- and cross-context components, and ii) approximating the cross-context component with canonical representations.

3.1 Term Score Factorization

COIL-tok implicitly models two distinct aspects of term match scoring: *intra-context importance*, which measures the importance of a term (q_i or d_j) to its own text (q or d), and *cross-context similarity*, which measures whether matching terms q_i and d_j are used in a similar context and require actual interaction at retrieval time. As shown in previous term-weighting systems and COIL model variants (e.g., UniCOIL [16]), the term importance component can be effectively represented with a scalar. A more critical question lies in the capacity and cost of representing term semantics and calculating query-document similarity.

COILcr explicitly factorizes COIL's contextualized token representations into a scalar *term weight* value and a *term semantic representation* vector for each term, using separate linear projection layers:

$$\begin{aligned} w_{d_j} &= \phi_w(LM(d,j)) \\ \mathbf{v}_{d_j} &= \phi_v(LM(d,j)) \\ \hat{\mathbf{v}}_{d_j} &= \frac{\mathbf{v}_{d_j}}{||\mathbf{v}_{d_j}||} \end{aligned}$$

where w_{d_j} is a non-negative value denoting term weight, and $\hat{\mathbf{v}}_{d_j}$ is a normalized vector denoting term semantics.

COILcr uses the same language model to encode query and document terms. The factorized contextualized exact match score between overlapping terms $q_i = d_j$ is defined as:

$$\begin{aligned} s(q_i, d_j) &= w_{q_i} w_{d_j} cos(\mathbf{v}_{q_i}, \mathbf{v}_{d_j}) \\ &= w_{q_i} w_{d_j} \hat{\mathbf{v}}_{q_i}^{\intercal} \hat{\mathbf{v}}_{d_j} \end{aligned} \tag{1}$$

where w, v and s represents the weighting component, semantic similarity component and overall token matching score respectively. Equation 1 can be viewed as a factorization of COIL's dot-product scoring function to an equivalent cosine similarity form. It explicitly decouples term weighting to enable more direct analysis and approximation of term semantics.

The exact overall score between a query q and document d is the sum of all lexical matching scores of overlapping terms. Following COIL, we train COILcr with an NLL loss defined on query q, document set $\{d^+, d_1^-, d_2^-, ..., d_{n-1}^-\}$, and the scoring function.

$$\begin{aligned} \mathcal{S}_e(q,d) &= \sum_{q_i \in \mathcal{V}_q \cap \mathcal{V}_d} \max_{d_j = q_i} \; s(q_i, d_j) \\ &= \sum_{q_i \in \mathcal{V}_q \cap \mathcal{V}_d} \max_{d_j = q_i} w_{q_i} w_{d_j} \hat{\mathbf{v}}_{q_i}^{\intercal} \hat{\mathbf{v}}_{d_j} \\ \mathcal{L}_{NLL} &= -\log \frac{\exp(\mathcal{S}_e(q,d^+))}{\exp(\mathcal{S}_e(q,d^+)) + \sum_{i=1}^{n-1} \exp(\mathcal{S}_e(q,d_i^-))} \end{aligned}$$

3.2 Approximate Term Semantic Interaction

The main additional cost of COIL compared to other lexical exact-match systems lies in storing a unique vector for each document term occurrence during indexing, and having to compute vector products $\mathbf{v}_{q_i}^{\intercal} \mathbf{v}_{d_j}$ or $\hat{\mathbf{v}}_{q_i}^{\intercal} \hat{\mathbf{v}}_{d_j}$ for each document term occurrence at retrieval time. COILcr mainly focuses on approximating the vector product by reducing the space of vocabulary representations. For a term t, instead of using unique vectors for every term occurrence, COILcr selects a fixed set of semantic *canonical representations* C_t after the encoding stage, and maps each term semantic representation to its closest vector in C_t.

$$\mathbf{c}_{d_j} = \underset{c \in \mathbf{C_t}}{\arg\max} \ \cos(\hat{\mathbf{v}}_{d_j}, c)$$

where $\mathbf{c}_{d_j}$ can be viewed as an approximate representation of the original term $\mathbf{v}_{d_j}$. At retrieval time, $\mathbf{c}_{d_j}$ is used to calculate an approximated term matching score and the final document score.

$$s_c(q_i, d_j) = w_{q_i} w_{d_j} \hat{\mathbf{v}}_{q_i}^{\mathsf{T}} \mathbf{c}_{d_j}$$
$$\mathcal{S}_c(q, d) = \sum_{q_i \in q \cap d} \max_{d_j = q_i} \ w_q w_d \ \hat{\mathbf{v}}_{q_i}^{\mathsf{T}} \mathbf{c}_{d_j}$$

Mapping unique term occurrence representations to canonical term occurrence representations reduces the storage cost of each individual term occurrence from a unique $|d|$-dim vector to just its term weight w and the index of its canonical representation. At retrieval time, instead of calculating $\hat{\mathbf{v}}_{q_i}^{\mathsf{T}} \hat{\mathbf{v}}_{d_j}$ for each term occurrence d_j, COILcr only needs to calculate $\hat{\mathbf{v}}_{q_i}^{\mathsf{T}} \mathbf{c}$ for each CR $\mathbf{c} \in C_t$. The actual term representation scoring is reduced to a lookup operation of $\hat{\mathbf{v}}_{q_i}^{\mathsf{T}} \mathbf{c}_{d_j}$ from the set of candidate scores.

Canonical semantic representations C_t can be generated in varied ways. COILcr generates them using weighted spherical k-means clustering [3]. For each term, it iterates to optimize

$$\mathbf{F}_t = \sum_{d_j = t} w_{d_j} cos(\hat{\mathbf{v}}_{d_j}, \mathbf{c}_{d_j})$$

where $\mathbf{F}_t$ is a weighted sum of cosine similarity between $\hat{\mathbf{v}}_{d_j}$ and $\mathbf{c}_{d_j}$. This is aligned with the scoring function of COILcr.

The number of canonical representations $|C_t|$, or the number of clusters, directly determines the granularity of term semantics and the degree of approximation. In this work we experiment with three cluster selection methods.

– Constant: A fixed number of clusters $|C|$ is generated for all terms.
– Dynamic: The cluster size is determined dynamically based on a *clustering error threshold*. Given an error threshold ϵ and a set of candidate cluster sizes $\{\mathbf{K}^d\}$, for each term the minimum cluster size $k_t^d \in \{\mathbf{K}^d\}$ is selected such that the clustering error $E_t = 1 - \frac{1}{|d_t|}\mathbf{F_t}$ falls below ϵ.
– Universal: Following previous work [22], we include a separate experiment where all terms share a fixed set of universal canonical representations. The centroids are generated by randomly sampling term representations in the entire corpus and performing clustering.

We perform detailed analysis of the effect of cluster size selection in Section 5.2. In the sections below, we refer to COILcr variants that perform clustering as COILcr-t,k where t is the type of clustering strategy (c/d/u for constant, dynamic, universal) and k is the respective parameter (cluster size $|C|$ for constant and universal, error threshold ϵ for dynamic). When no clustering approximation is performed, COILcr is equivalent to COIL-tok with factorized term scoring. We refer to this model variant as COILcr-∞ (infinite clusters).

4 Experimental Methodology

Implementation COILCR mostly follows COIL's implementation[2] and training settings, using a default token representation with $d_v = 32$ dimensions. We analyze the effect of token representation dimension in Section 5. All COILCR variants are trained for 5 epochs with a batch size of 8 queries and 8 documents (1 positive, 7 negative) per query. At indexing time, we randomly sample representations and perform spherical k-means clustering with Faiss [13]. We experiment with $k \in \{1, 4, 16, 64, 256, 1024\}$ clusters for constant and dynamic cluster generation, error thresholds $\epsilon \in \{0.05, 0.1, 0.15, 0.2, 0.25\}$ for dynamic cluster generation, and $k \in \{256, 1024, 4096\}$ for universal cluster generation.

Extensions COILCR does not perform expansion or remapping of original terms, but can be used with document expansion systems such as DocT5Query [19]. We also experiment with model initialization using coCondenser[7], a deep LM trained on retrieval-related tasks, as this has been effective in prior work [4,22].

Experiments We train our model on the MSMARCO passage dataset [17] and report model performance on MSMARCO dev queries and TREC DL 2019 manual queries. We report MRR@10 and recall@1000 for MSMARCO dev evaluation, and report NDCG@10 and recall@1000 for TREC DL queries. We mainly focus our comparison to previous COIL-based lexical exact match retrieval systems COIL-tok and its term-weighting variant UniCOIL. We also train and report results for UniCOIL and COIL-tok baselines with coCondenser initialization and DocT5Query augmentation. We additionally report the performance of two related retrieval systems: (1) COIL-full, a hybrid retriever that additionally utilizes a dense scoring component, and (2) SPLADE [5], an end-to-end term weighting-based lexical retrieval system with vocabulary expansion.

5 Experiments

In this section, we discuss the retrieval performance of COILCR and the effect of its components. We first separately analyze the effectiveness of explicit score factorization and post-hoc CR-based approximation. We further perform a quantitative analysis on the two main factors of COILCR's effectiveness-efficiency tradeoff: the vector representation dimension and the post-hoc approximation. We finally look into the semantic information of canonical representations through domain-transfer experiments and analysis.

5.1 Passage Retrieval Effectiveness

We first report the passage retrieval performance of COILCR-∞ variants on MSMARCO in Table 1. Under the same training settings, COILCR-∞ achieves very

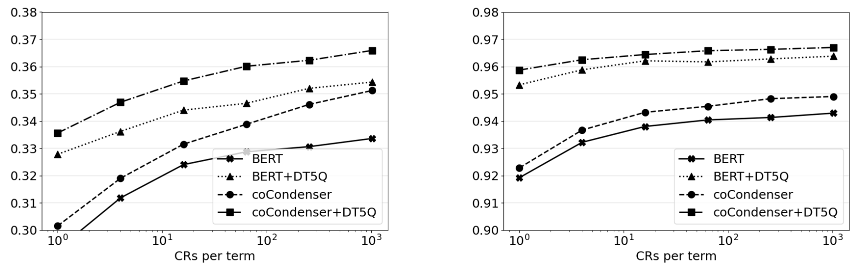
[2] https://github.com/luyug/COIL

Table 1: Passage retrieval performance for COILcr on MSMARCO. Baselines labeled with * were retrained. We perform significance testing for COILcr variants with coCondenser initialization. Under the same training settings, † denotes equal or better performance compared to COIL-tok and ‡ denotes better performance compared to UniCOIL.
† TOST testing with $\alpha = 5\%$ and equivalence bound of ± 0.005.
‡ Paired t-test with $\alpha = 5\%$.

Model		MSMARCO dev		Trec DL 2019	
Retriever	Init.	MRR@10	R@1000	NDCG@10	R@1000
Lexical retrievers w/o implicit vocabulary expansion					
UniCOIL	BERT	0.320	0.922	0.652	-
UniCOIL + DocT5Q	BERT	0.351	-	0.693	-
COIL-tok	BERT	0.341	0.949	0.660	-
UniCOIL*	coCondenser	0.328	0.929	0.646	0.778
UniCOIL + DocT5Q*	coCondenser	0.357	0.961	0.702	0.823
COIL-tok*	coCondenser	0.353	0.949	0.692	0.801
COIL-tok + DocT5Q*	coCondenser	0.365	0.967	0.707	0.833
Hybrid systems or lexical retrievers with implicit expansion					
COIL-full	BERT	0.355	0.963	0.704	-
COIL-full	coCondenser	0.374	0.981	-	-
SPLADE	BERT	0.322	0.955	0.665	0.813
COILcr-∞	BERT	0.341	0.944	0.673	0.787
COILcr-∞ + DocT5Q	BERT	0.358	0.964	0.692	0.830
COILcr-∞	coCondenser	0.355†‡	0.951†‡	0.717	0.794
COILcr-∞ + DocT5Q	coCondenser	0.370†‡	0.968†‡	0.711	0.832
COILcr-c256	BERT	0.331	0.941	0.676	0.784
COILcr-c256 + DocT5Q	BERT	0.352	0.963	0.698	0.831
COILcr-c256	coCondenser	0.346‡	0.948†‡	0.704	0.797
COILcr-c256 + DocT5Q	coCondenser	0.362‡	0.966†‡	0.714	0.836

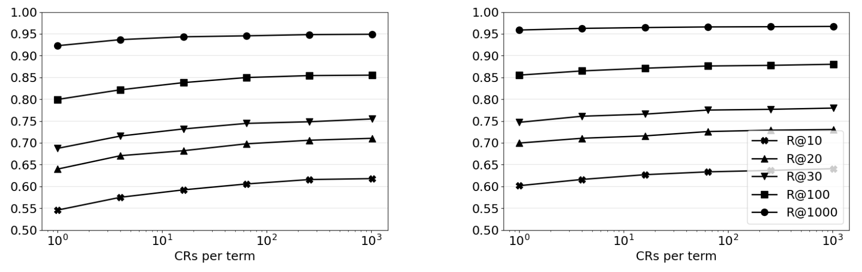
similar accuracy and Recall compared to COIL-tok. This demonstrates the extra capacity of modeling term semantics, and that COILcr's score factorization step does not limit such model capacity by itself.

Performing coCondenser initialization and DocT5Query augmentation improves the retrieval performance of all COILcr variants with different effects, as expected. Initialization with coCondenser, a system pretrained on the MSMARCO dataset and on a dense retrieval-related task, also helps lexical-only retrieval systems COIL-tok and COILcr learn higher quality term representations and more accurate matching signals, leading to improvement in model accuracy. On the other hand, COIL-based models do not implicitly resolve the vocabulary mismatch between the query and document. Under comparable training setups, COILcr-∞ and COIL-tok systems outperform SPLADE on accuracy at top positions (MRR@10 and NDCG@10 on respective datasets) but underperform on recall@1000. The addition of DocT5Query augmentation introduces explicit document expansion which leads to better overall performance, especially for Recall@1000 (0.95 → 0.97 for MSMARCO dev, 0.79 → 0.83 for Trec DL 2019). Specifically, on the Trec DL 2019 queries, UniCOIL achieves close performance to COILcr and COIL-tok with DocT5Query augmentation.



(a) MRR@10 (b) R@1000

Fig. 1: Passage retrieval performance on MSMARCO-dev for COILcr model variants with different degrees of approximation and different training setup.



(a) COILcr (b) COILcr+DocT5Q

Fig. 2: Recall at different depths (Recall@k) for COILcr model variants with different degree of approximation ($|C_t|$). Models are initialized with coCondenser.

After performing clustering with $|C_t| = 256$ CRs per term, we observe only a slight drop in MRR@10 and Recall. To further explore how post-hoc approximation affects COILcr's retrieval performance, we report the MRR@10 and Recall@1000 of COILcr on MSMARCO dev queries with different CR size $|C_t|$ in Figure 1, and the change in Recall at different depths with different $|C_t|$ in Figure 2. Under all training settings, the degree of approximation mainly affects the *precision* of lexical exact match signals and documents at the top of the ranking. It has particularly little impact on recall at lower positions, where the more critical bottleneck is vocabulary mismatch and sufficient lexical exact match signals do not exist.

Table 2: Passage retrieval accuracy and storage cost of COIL_{CR} with varying numbers of representation dimensions and CRs per term. Models initialized with coCondenser. † and ‡ respectively denotes equal or better performance compared to COIL-tok, and better performance compared to UniCOIL.

Model		MSMARCO dev		Storage (GB)		
		MRR@10	R@1000	CR Index	Inv. Index	Total
COIL-tok		0.353	0.949	n/a	45	45
UniCOIL		0.328	0.929	n/a	4.8	4.8
COILCR:						
32	∞	$0.355^{\dagger\ddagger}$	$0.951^{\dagger\ddagger}$	n/a	55	55
16	∞	$0.350^{\ddagger}$	$0.950^{\dagger\ddagger}$	n/a	34	34
8	∞	$0.350^{\ddagger}$	$0.946^{\dagger\ddagger}$	n/a	21	21
4	∞	$0.345^{\ddagger}$	$0.941^{\ddagger}$	n/a	14	14
32	c256	$0.346^{\ddagger}$	$0.948^{\dagger\ddagger}$	0.7	5.5	6.2
16	c256	$0.343^{\ddagger}$	$0.947^{\dagger\ddagger}$	0.4	5.4	5.8
8	c256	$0.349^{\ddagger}$	$0.945^{\ddagger}$	0.2	5.5	5.7
4	c256	$0.343^{\ddagger}$	$0.941^{\ddagger}$	0.1	5.4	5.4
32	c256	$0.346^{\ddagger}$	$0.948^{\ddagger}$	0.7	5.5	6.2
32	c64	$0.340^{\ddagger}$	$0.946^{\ddagger}$	0.2	5.4	5.6
32	c16	0.331	$0.943^{\ddagger}$	0.1	5.2	5.3
32	c4	0.320	$0.938^{\ddagger}$	0.02	5.1	5.1
32	c1	0.302	0.923	<0.01	4.9	4.9

5.2 Balancing Model Efficiency

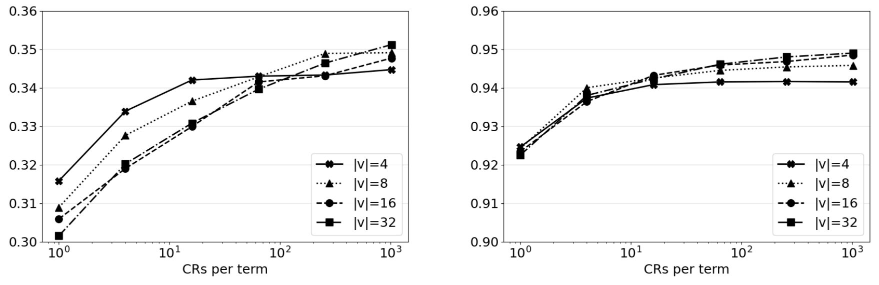
Next, we examine the effectiveness-efficiency tradeoff of COILCR and its two main factors, the number of term representation dimensions and the degree of approximation from original term representations to canonical representations.

Table 2 shows the model accuracy and storage cost of COILCR on the MS-MARCO passage dataset with varying representation sizes d_v and CRs per term $|C_t|$. By reducing each inverted index entry to a term weight and a CR index, COILCR significantly lowers the storage cost of the COIL index. The content and storage cost of the inverted index entry remains the same regardless of representation dimension changes.

All COILCR-∞ systems outperform the UniCOIL baseline where d_v=1. We further report the performance of COILCR variants with different representation dimensions d_v in Figure 3. Higher dimension representations lead to a higher ceiling in model accuracy, but require more CRs per term to reach such performance. On the other hand, the overall difference in Recall@1000 for different d_v and different CR size becomes relatively small after very coarse term semantic modeling ($|C_t| > 16$). This may be beneficial in Recall-oriented settings such as first-stage ranking in a reranking pipeline, when a lower d_v and $|C_t|$ reduces run time and storage cost while not affecting the overall performance of the system.

Table 3: Passage retrieval accuracy and retrieval cost of COILcr with different CR generation strategies. c/d/u respectively denotes the constant, dynamic and universal clustering approaches, as discussed in Section 3.2. Models initialized with coCondenser.

Model	Model Performace		Run Cost	
	MRR@10	R@1000	CR Storage	Avg. Ops.
c1024	0.351	0.949	2.5	1024
c256	0.346	0.948	0.77	256
c64	0.340	0.946	0.2	64
c16	0.331	0.942	0.06	16
c4	0.320	0.938	0.02	4
c1	0.302	0.923	<0.01	1
d0.05	0.349	0.949	2.3	997.4
d0.1	0.349	0.948	1.1	591.5
d0.15	0.344	0.947	0.27	152.5
d0.2	0.337	0.945	0.1	41.34
d0.25	0.330	0.942	0.05	14.76
u4096	0.346	0.946	<0.01	2740
u1024	0.339	0.945	<0.01	823
u256	0.336	0.943	<0.01	233
Ctok-c1024	0.339	0.946	-	-
Ctok-c256	0.329	0.944	-	-
Ctok-c64	0.309	0.939	-	-



(a) MRR@10 (b) R@1000

Fig. 3: Passage retrieval performance on MSMARCO-dev for COILcr model variants with different representation dimensions.

5.3 Canonical Representation Analysis

In this section, we take a deeper look into the construction process and properties of canonical representations in COILcr. We first compare different CR selection strategies discussed in Section 3.2, and report model performance on MSMARCO in Table 3. As discussed in Section 3.2, for a query term at retrieval

Table 4: Zero-shot retrieval accuracy (nDCG@10) on the BEIR benchmark. Co-Condenser initialization and DocT5Q augmentation were applied for all models. Best performance of each dataset is underlined.

Corpus	UniCOIL	COILcr-∞	COILcr-c256	COILcr-c256-tr
ArguAna	0.365	0.342	0.339	0.341
C-FEVER	0.178	0.186	0.188	0.188
DBPedia	0.360	0.378	0.380	0.376
FEVER	0.778	0.782	0.793	0.797
FiQA	0.293	0.310	0.303	0.297
HotpotQA	0.662	0.683	0.679	0.675
NFCorpus	0.336	0.338	0.338	0.336
NQ	0.446	0.485	0.483	0.477
Quora	0.732	0.773	0.762	0.750
SCIDOCS	0.150	0.153	0.154	0.151
SciFact	0.696	0.698	0.699	0.697
T-COVID	0.739	0.735	0.739	0.745
Touche2020	0.279	0.287	0.289	0.292

time, COILcr only performs vector product with its canonical representations instead of the representation of every term occurrence. In addition to CR index storage cost, we report the average number of retrieval-time *vector product operations*, or the average number of canonical representations a query term matches, to compare the computational cost between COILcr variants. Compared to term-specific CR selection, universal CR selection introduces much less storage cost, but naturally requires a larger set of CRs to preserve the semantics of *all terms*, and leads to extra operations at retrieval time. The two term-specific CR selection approaches have similar performance trends, as they require similar storage and operation costs to achieve the same level of performance.

To investigate the effect of factorizing term weight and term semantics, we additionally perform a side experiment where we directly generate canonical representations from COIL-tok term representations via k-means clustering. We denote this retrieval approach as Ctok-ck and report performance in Table 3. Compared to COILcr, the canonical representations generated from COIL-tok need to preserve extra information of the representation *norm*, which affects distance and loss calculation and leads to inefficient K-means clustering. Thus, this approach naturally requires much more CRs per term to reach the same retrieval performance as COILcr.

Additionally, to investigate the robustness of the COILcr system and the CR approximation approach, we take COILcr trained on MSMARCO and perform a *zero-shot* retrieval experiment on the BEIR [23] benchmark, which consists of datasets covering a wide range of different domains. We also introduce an extra COILcr variant, denoted as COILcr-tr, where we also directly transfer the CRs generated from MSMARCO representations, instead of generating from the new dataset. We report performance results on 13 datasets in the

BEIR benchmark in Table 4. We observe that COILcr-∞ maintains its extra model capacity over UniCOIL, with larger than 3% gains on 7 of 13 datasets. the only dataset where COILcr clearly underperforms UniCOIL is ArguAna, which involves retrieval of counterarguments given a query argument, and is very different from classic web search settings. Moreover, across all datasets, the model accuracy of COILcr and COILcr-tr remains similar and close to the performance of COILcr-∞. This demonstrates the robustness of the CR approximation approach with sufficient clusters and suggests that the main bottleneck for COILcr in the zero-shot retrieval setting lies in the language model base, at the step of term representation generation.

6 Conclusion and Future Work

This paper investigates the model capacity and runtime cost of COIL-style lexical retrievers. We present COILcr, an extension to COIL which factorizes term representations into weighting and semantic components. At indexing time, COILcr constructs semantic canonical representations to approximate term semantics and precise matching between query and document terms, leading to reduced index storage and retrieval runtime cost.

Without approximation, COILcr-∞ maintains the model capacity of COIL-tok and consistently outperforms UniCOIL. Performing CR-based approximation for COILcr only slightly affects model accuracy, but drastically reduces the inverted index storage cost by 90% while also transforming most run-time vector product operations to a simple lookup operation.

Our experiments examine the effectiveness-efficiency balance of COILcr, and discuss the effects of different term representation sizes and clustering heuristics on model performance. We find that model accuracy is more prone to error from approximation, while consistent Recall performance can be achieved with very coarse term semantics. Experiments under different approximation and retrieval settings further demonstrate the robustness of the CR approximation approach.

Throughout this work, we observe the necessity of modeling term semantics in lexical exact match retrieval, as well as the potential of very efficiently doing so. In this paper, we utilize spherical clustering as a simple post-hoc approach for CR generation and note the possibility of finding improved methods to build canonical representation sets which reflect term senses. We hope this is an encouraging step towards building both effective and efficient lexical retrieval models and indexes in the future.

References

1. Dai, Z., Callan, J.: Context-aware document term weighting for ad-hoc search. In: Proceedings of The Web Conference 2020. pp. 1897–1907 (2020)
2. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)

3. Dhillon, I.S., Modha, D.S.: Concept decompositions for large sparse text data using clustering. Machine learning **42**(1), 143–175 (2001)
4. Formal, T., Lassance, C., Piwowarski, B., Clinchant, S.: Splade v2: Sparse lexical and expansion model for information retrieval. arXiv preprint arXiv:2109.10086 (2021)
5. Formal, T., Piwowarski, B., Clinchant, S.: 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 (2021)
6. Gao, L., Callan, J.: Condenser: a pre-training architecture for dense retrieval. arXiv preprint arXiv:2104.08253 (2021)
7. Gao, L., Callan, J.: Unsupervised corpus aware language model pre-training for dense passage retrieval. arXiv preprint arXiv:2108.05540 (2021)
8. Gao, L., Dai, Z., Callan, J.: COIL: revisit exact lexical match in information retrieval with contextualized inverted list. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021. pp. 3030–3042. Association for Computational Linguistics (2021). https://doi.org/10.18653/v1/2021.naacl-main.241, `https://doi.org/10.18653/v1/2021.naacl-main.241`
9. Hofstätter, S., Althammer, S., Schröder, M., Sertkan, M., Hanbury, A.: Improving efficient neural ranking models with cross-architecture knowledge distillation. arXiv preprint arXiv:2010.02666 (2020)
10. Hofstätter, S., Lin, S.C., Yang, J.H., Lin, J., Hanbury, A.: Efficiently teaching an effective dense retriever with balanced topic aware sampling. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 113–122 (2021)
11. Izacard, G., Caron, M., Hosseini, L., Riedel, S., Bojanowski, P., Joulin, A., Grave, E.: Towards unsupervised dense information retrieval with contrastive learning. arXiv preprint arXiv:2112.09118 (2021)
12. Izacard, G., Petroni, F., Hosseini, L., De Cao, N., Riedel, S., Grave, E.: A memory efficient baseline for open domain question answering. arXiv preprint arXiv:2012.15156 (2020)
13. Johnson, J., Douze, M., Jégou, H.: Billion-scale similarity search with GPUs. IEEE Transactions on Big Data **7**(3), 535–547 (2019)
14. Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., Yih, W.t.: Dense passage retrieval for open-domain question answering. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 6769–6781 (2020)
15. Khattab, O., Zaharia, M.: Colbert: Efficient and effective passage search via contextualized late interaction over bert. In: Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval. pp. 39–48 (2020)
16. Lin, J., Ma, X.: A few brief notes on deepimpact, coil, and a conceptual framework for information retrieval techniques. CoRR **abs/2106.14807** (2021), `https://arxiv.org/abs/2106.14807`
17. Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., Deng, L.: Ms marco: A human generated machine reading comprehension dataset. In: CoCo@ NIPS (2016)
18. Nogueira, R., Cho, K.: Passage re-ranking with bert. arXiv preprint arXiv:1901.04085 (2019)

19. Nogueira, R., Lin, J., Epistemic, A.: From doc2query to docttttquery. Online preprint **6** (2019)
20. Qu, Y., Ding, Y., Liu, J., Liu, K., Ren, R., Zhao, W.X., Dong, D., Wu, H., Wang, H.: Rocketqa: An optimized training approach to dense passage retrieval for open-domain question answering. arXiv preprint arXiv:2010.08191 (2020)
21. Robertson, S., Zaragoza, H., et al.: The probabilistic relevance framework: Bm25 and beyond. Foundations and Trends® in Information Retrieval **3**(4), 333–389 (2009)
22. Santhanam, K., Khattab, O., Saad-Falcon, J., Potts, C., Zaharia, M.: Colbertv2: Effective and efficient retrieval via lightweight late interaction. arXiv preprint arXiv:2112.01488 (2021)
23. Thakur, N., Reimers, N., Rücklé, A., Srivastava, A., Gurevych, I.: Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models. arXiv preprint arXiv:2104.08663 (2021)
24. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
25. Xiong, L., Xiong, C., Li, Y., Tang, K.F., Liu, J., Bennett, P., Ahmed, J., Overwijk, A.: Approximate nearest neighbor negative contrastive learning for dense text retrieval. arXiv preprint arXiv:2007.00808 (2020)
26. Yamada, I., Asai, A., Hajishirzi, H.: Efficient passage retrieval with hashing for open-domain question answering. arXiv preprint arXiv:2106.00882 (2021)
27. Zhan, J., Mao, J., Liu, Y., Guo, J., Zhang, M., Ma, S.: Jointly optimizing query encoder and product quantization to improve retrieval performance. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management. pp. 2487–2496 (2021)
28. Zhan, J., Mao, J., Liu, Y., Guo, J., Zhang, M., Ma, S.: Learning discrete representations via constrained clustering for effective and efficient dense retrieval. In: Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining. p. 1328–1336. WSDM '22, Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3488560.3498443, `https://doi.org/10.1145/3488560.3498443`
29. Zhao, T., Lu, X., Lee, K.: 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. Association for Computational Linguistics, Online (Jun 2021). https://doi.org/10.18653/v1/2021.naacl-main.47, `https://aclanthology.org/2021.naacl-main.47`