DOCUMENT OPTIMIZATION FOR BLACK-BOX RETRIEVAL VIA REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative large language models (LLMs) are increasingly used as inference-time components in retrieval pipelines, for tasks such as query rewriting and document reranking. However, these online approaches place costly autoregressive computation directly on the latency-critical retrieval path. We explore an alternative axis: using LLMs to improve documents instead, rewriting them into better representations and shifting computation offline. Yet producing a useful document rewrite is not straightforward: retrieval is inherently discriminative, so an effective rewrite must make a document more similar to relevant queries than competing candidates under the retriever's notion of similarity. We therefore formulate document transformation as an optimization problem, directly training an LLM or VLM to produce rewrites that improve retrieval. Our approach, DocOpt, uses GRPO with retriever ranking improvements as rewards, requires only black-box access to retrieval ranks, and applies across single-vector, multi-vector, and lexical retrievers. We evaluate zero-shot LLM rewriting and DocOpt on code and visual retrieval tasks, finding that document rewriting can improve retrieval and that optimizing rewrites yields further gains. For example, OpenAI text-embedding-3-small achieves 58.35 nDCG@5 on average with direct retrieval; zero-shot rewriting improves this to 60.83 with GPT-5.4-mini, 63.75 with Claude Haiku 4.5, and 64.23 with Qwen3. DocOpt further improves performance to 67.94, surpassing the 6.5× more expensive text-embedding-3-large retriever at 66.15.

1 INTRODUCTION

Large language models (LLMs) have become central to modern information retrieval. While encoder models serve as the representational backbone of neural retrievers (Lee et al., 2019; Karpukhin et al., 2020; Khattab & Zaharia, 2020), generative LLMs can further augment retrieval pipelines, for example by rewriting queries (Xu & Croft, 2000; Gao et al., 2023; Hsu et al., 2024) or reranking a limited set of candidate documents (Sun et al., 2023; Qin et al., 2024). These applications, however, are constrained by latency and serving cost, as they place expensive LLM inference directly on the latency-critical retrieval path.

An alternative is to apply LLMs on the document side, where computation can be performed offline before indexing (Nogueira et al., 2019; Weller et al., 2024; Kuo et al., 2025). Rather than invoking an LLM for every query, each document can be transformed by an LLM for improved retrieval, allowing the cost to be amortized across all future queries while preserving a lightweight retrieval pipeline at inference time. However, naively transforming documents with generative LLMs does not reliably improve retrieval and can even degrade performance (Gospodinov et al., 2023; Weller et al., 2024; Kuo et al., 2025). LLMs are optimized with generative objectives, whereas retrieval is inherently discriminative: the value of a document rewrite lies in making the document more distinguishable from competing candidates for its target queries under the retriever's notion of similarity.

In this paper, we introduce ***Doc**ument **Opt**imization*, where we learn to rewrite documents into retrieval-oriented surrogates, under a specific retriever (Figure 1), and show that it can be a powerful, flexible tool for improving retrieval in a variety of settings.

Learning to rewrite documents is challenging because the objective is defined over discrete text, and the quality of a rewrite is only revealed through its downstream retrieval behavior. We therefore formulate document optimization as a reinforcement learning problem. In this mode, we treat an

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

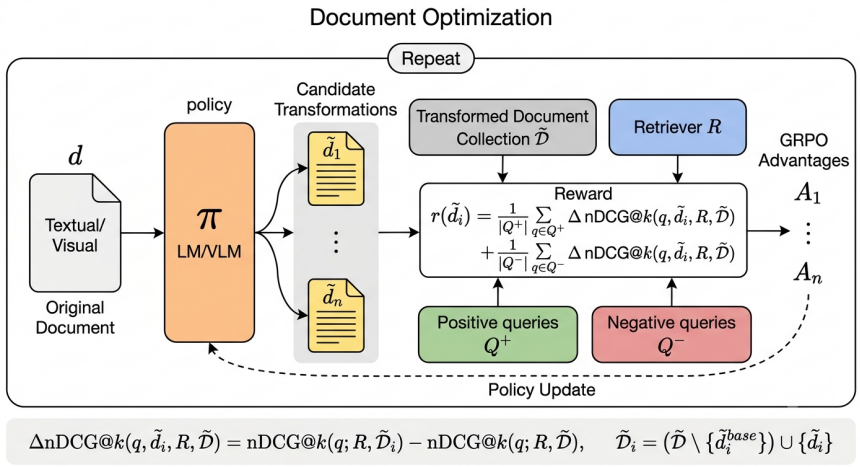

Figure 1: Document Optimization. At each iteration, a policy π generates candidate transformations $\tilde{d}_i$ for a document d. Each transformation is evaluated under a fixed collection $\tilde{\mathcal{D}}$ and retriever R using positive and negative queries, Q^+ and Q^-. Rewards are computed via the counterfactual $\Delta\mathrm{nDCG@}k$, capturing retrieval-quality gains on both positive and negative query sets. These are converted into GRPO advantages to update the policy. Through training, the policy learns to generate transformations which improve rankings under the retriever's representation space. Bottom: definition of $\Delta\mathrm{nDCG@}k$.

instruction-tuned language model (LM) or vision language model (VLM) as a policy over document rewrites. Rewards are defined counterfactually: for a proposed rewrite, we measure how retrieval would change if we replaced the original document in the index while all other documents were held fixed. Effective rewrites improve ranking of the document for relevant queries and suppress spurious matches to irrelevant ones. We optimize this policy using Group Relative Policy Optimization (GRPO) (Shao et al., 2024). For each document, the policy samples candidate rewrites and receives rewards based on their counterfactual retrieval utility, gradually learning to produce rewrites that are better aligned with the retriever's representation space (Figure 2).

This method applies in black-box settings and is compatible with single-vector (Lee et al., 2019; Karpukhin et al., 2020), multi-vector (Khattab & Zaharia, 2020; Santhanam et al., 2022b), and lexical retrievers (Robertson & Zaragoza, 2009) alike, since the learning signal depends only on retriever rankings, rather than gradients or access to retriever internals. We evaluate our approach on two retrieval settings: visual document retrieval, where images of PDF pages are converted into textual surrogates for text retrieval, and code retrieval, where code snippets are rewritten into retrieval-optimized textual representations. We test a diverse pool of retrievers, both open-weight and closed-weight, and compare our approach to using proprietary LLMs like GPT and Claude.

Across both visual document retrieval and code retrieval, we find that transforming documents with generative LLMs improves retrieval, and that optimizing these transformations with DocOpt yields additional gains. In the black-box setting, document optimization yields large consistent gains over direct retrieval and zero-shot transformation, for both closed-weight and open-weight models. For example, applying document optimization to OpenAI text-embedding-3-small improves nDCG@5 from 58.35→67.94 on average, both on code (68.63→78.23) and VDR (48.08→57.65). DocOpt outperforms both commercial LLM baselines, GPT-5.4-mini at 60.83 and Claude Haiku 4.5 at 63.75, and applying the 6.5× more expensive text-embedding-3-large at 66.15. We find that gains extend across families and architectures: Qwen3-Embedding-4B, improves from 63.97 to 71.17 on average, while Jina-ColBERT-V2 sees a substantial boost from 56.31 to 65.95. In the open-weight retriever setting, document optimization is competitive with fine-tuning on most settings, and combining both appears especially effective for multi-vector retrievers: Jina-ColBERT-V2 increases from 56.31 to 69.15, while finetuning alone only reaches 64.14.

Notably, all of these improvements are achieved entirely offline, preserving inference-time efficiency.

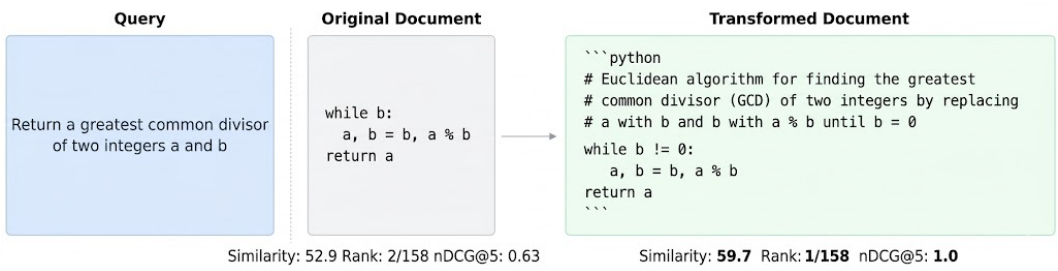

Figure 2: Example of document transformation improving retrieval quality. A HumanEval (MTEB) query (left) is matched against a labeled code document: the original (middle) and its transformed version (right). The transformation adds natural language comments, introduces minor code edits (e.g., `while b` → `while b != 0`), and wraps the code in a `python` block. Under the OpenAI text-embedding-3-small model, similarity increases (52.9 → 59.7), rank improves (2 → 1), and nDCG@5 rises (0.63 → 1.0).

2 Background

Passage Retrieval. Let $D = (d_1, ..., d_n)$ be a set of documents, and let q denote a textual query. The retrieval task is to return the top-k items of D that are most relevant[1] to q, where k is a hyperparameter. Retrieval quality is commonly evaluated using Recall@k and nDCG@k. Recall@k measures how many relevant documents appear in the top-k results, and is particularly useful for evaluating first-stage retrieval in multi-stage pipelines. In this work we adopt nDCG@k as the primary metric, since it additionally accounts for documents ranking in the top-k by rewarding higher placements (see Appendix A.1 for formal definitions).

Bi-Encoders. Modern retrieval systems often rely on bi-encoders (Reimers & Gurevych, 2019; Karpukhin et al., 2020; Khattab & Zaharia, 2020), which map queries and documents independently into a shared representation space and score relevance through a non-parametric similarity function. These models are typically initialized from pretrained bidirectional LMs (Devlin et al., 2019) and adapted for retrieval using noise contrastive estimation (Gutmann & Hyvärinen, 2010; van den Oord et al., 2019) over labeled queries and documents. A key property of bi-encoders is that document representations can be precomputed offline. At inference time, the system encodes only the query and efficiently searches a prebuilt index (Jégou et al., 2011; Johnson et al., 2017; Malkov & Yashunin, 2020; Santhanam et al., 2022a) to retrieve the top-k highest-scoring documents.

Foundation Models as Policies. An LM or a VLM can be described as a stochastic policy that maps from an input state (prompt) to a distribution over actions (next-token predictions), where a sequence of such predictions forms a trajectory. Policy gradient methods (Williams, 1992; Sutton et al., 1999; Schulman et al., 2017) have been effective at improving the performance of LMs on various tasks (Ziegler et al., 2020; Ouyang et al., 2022; Shao et al., 2024). More recently, Group Relative Policy Optimization (GRPO) (Shao et al., 2024) and its variants (Liu et al., 2025b; Yu et al., 2025) have emerged as an effective approach, leveraging relative reward comparisons between action trajectories for variance reduction without requiring an explicit value function. In our approach, the policy model can be any generative model that maps inputs into the appropriate representation space for the retriever.

3 Method

We assume a supervised setting with access to annotated sets of positive and negative queries for each document d_i, denoted Q_i^+ and Q_i^-.

Given a retriever R, our goal is to learn a policy π_ϕ (in this section we will refer to the document optimization model as a policy) that generates a rewritten document $\tilde{d}$ such that $\tilde{d}$ improves retrieval performance under R for queries relevant to d.

[1]Where relevance is defined by a binary label, typically via human annotation

We initialize π_ϕ from an instruction-tuned LM or VLM, conditioned on a fixed prompt p_{trans} that specifies the desired transformation (we provide the prompts used in our experiments in Appendix A.2). We then sample a transformed document from the policy:

$$\tilde{d} \sim \pi_\phi(\cdot \mid p_{trans}; d). \tag{1}$$

3.1 Counterfactual Retrieval Reward

To evaluate a rewrite $\tilde{d}_i \sim \pi_\phi(\cdot \mid p_{trans}; d_i)$, we measure its *counterfactual impact* on retrieval under R.

We first construct an initial rewritten collection by sampling a transformation for each document, yielding $\tilde{\mathcal{D}}$. For a given document d_i, we then define a counterfactual collection in which only its rewritten form is changed:

$$\tilde{\mathcal{D}}_i := (\tilde{\mathcal{D}} \setminus \{\tilde{d}_i^{\text{base}}\}) \cup \{\tilde{d}_i\}, \tag{2}$$

where $\tilde{d}_i^{\text{base}}$ denotes the baseline transformation of d_i in $\tilde{\mathcal{D}}$. Importantly, $\tilde{\mathcal{D}}$ is held fixed when evaluating the effect of replacing $\tilde{d}_i^{\text{base}}$ with $\tilde{d}_i$.

Given a query q, let $\text{nDCG@}k(q; R, \mathcal{D})$ denote the retrieval quality at cutoff k (in our experiments we use $k = 5$) when using retriever R over collection $\mathcal{D}$. We define the counterfactual change for a query as:

$$\Delta\text{nDCG@}k(q) := \text{nDCG@}k(q; R, \tilde{\mathcal{D}}_i) - \text{nDCG@}k(q; R, \tilde{\mathcal{D}}). \tag{3}$$

The reward aggregates these changes over positive and negative query sets Q_i^+ and Q_i^- associated with d_i:

$$r_i(\tilde{d}_i) = \underbrace{\frac{1}{|Q_i^+|} \sum_{q \in Q_i^+} \Delta\text{nDCG@}k(q)}_{\text{positive gain}} + \underbrace{\frac{1}{|Q_i^-|} \sum_{q \in Q_i^-} \Delta\text{nDCG@}k(q)}_{\text{negative gain}}. \tag{4}$$

The baseline transformation $\tilde{d}_i^{\text{base}}$ can be interpreted as the current sample from the policy, and the reward measures how alternative samples compare to it under a fixed collection. This isolates the effect of the transformation on ranking, encouraging improved retrieval for relevant queries while discouraging spurious matches for negative queries. We only take training steps for documents with non-empty positive and negative query sets, avoiding division by zero in the reward computation.

Reward Choice. The reward directly optimizes the target evaluation metric, $\text{nDCG@}k$, rather than relying on a proxy. It is agnostic to the underlying similarity function and applies uniformly across different retriever architectures. Notably, ranking-based rewards are sparse, as many document transformations yield identical scores when they do not affect the ranking. An alternative is to define the reward using query–document similarity, which provides a dense signal, but lacks the context of other documents in the index. While we find the ranking-based reward to perform better empirically, other formulations can also be effective under this framework. An empirical comparison is in Appendix D.

3.2 Policy Optimization

We optimize π_ϕ using Group Relative Policy Optimization (GRPO) (Shao et al., 2024). We provide the full details of the GRPO configuration we use in Appendix A.3. During each GRPO iteration, we sample a batch of documents $\mathcal{B} \subset D$ with annotated queries. For each document $d_i \in \mathcal{B}$, we generate G candidate transformations from π_ϕ, forming a group of candidate rewrites for that document. Each candidate is evaluated using the counterfactual retrieval reward against $\tilde{D}$, and we compute relative advantages within each group. We then perform a single policy update per batch, aggregating learning signals across all document-level groups. We use the temperature parameter to balance exploration and quality in the generated transformations. For index population, we use a low temperature to ensure stable, high-quality rewrites, while for generating the transformed collection during training, we use a higher temperature to encourage exploration. Finally, when training completes, we transform the entire document collection again using the learned policy and use the resulting corpus for retrieval. Computing rewards requires evaluating transformations against the retrieval index. However, this

is computationally prohibitive for large $|D|$ at every step. Instead, we only re-transform the full collection periodically every T_{refresh} steps, introducing a controlled degree of staleness.

3.3 Joint retriever-policy adaptation

Document optimization is not a replacement for retriever fine-tuning, which can be highly beneficial when retriever weights are available. We therefore consider jointly adapting the retriever and the policy. In a straightforward variant, R is periodically fine-tuned on the current transformed collection every T_{refresh} steps. This can also be viewed as a form of data augmentation, where the retriever is trained on a different set of document transformations at each iteration.

4 Experiments

4.1 Setup

Tasks. We evaluate our method on two retrieval settings: visual document retrieval (VDR) and code retrieval. VDR (Mathew et al., 2021b;a; Li et al., 2024; Zhu et al., 2022; Faysse et al., 2025) considers a collection of visually rich documents, including charts, images, and tables, paired with text queries grounded in the document content. We apply our method to perform image-to-text transformations of visual content. In this setting, the transformation policy maps each document image into a textual surrogate that is subsequently embedded by a text retriever. Code Retrieval assumes a corpus of code documents (e.g., functions or code snippets) paired with natural language queries describing their functionality. Here, the policy rewrites each code snippet into a retrieval-oriented textual representation (e.g., Figure 2), making the document easier to match with natural-language queries.

These domains are natural candidates for testing the value of LLMs for document-side adaptation. Converting PDFs to text is a common practice which already constitutes a form of document transformation. In code retrieval, code snippets can be enriched with comments or rewritten into more readable formats that could make them easier to retrieve. For evaluation, we use four datasets from ViDoRe2 (Macé et al., 2025) for VDR, and code retrieval tasks from MTEB (Muennighoff et al., 2023): HumanEvalRetrieval (Chen et al., 2021),MBPPRetrieval (Austin et al., 2021), and DS1000Retrieval (Lai et al., 2022).

Retrievers. For black-box retrievers, we use OpenAI's text-embedding-3-small and text-embedding-3-large through the OpenAI API. For open-weights models, we use Qwen3-Embedding-0.6B and Qwen3-Embedding-4B as single-vector retrievers, and Jina-ColBERT-V2 as a multivector one. We additionally evaluate BM25 as a lexical retriever.

Generative models. For visual document retrieval, we use Qwen3-VL-2B-Instruct (Qwen Team, 2025) to generate textual descriptions from document images. For code, we use Qwen3-4B-Instruct-2507 (Qwen Team, 2025). We additionally evaluate two proprietary LLMs, GPT-5.4 mini (OpenAI, 2026) and Claude Haiku 4.5 (Anthropic, 2025), as zero-shot document rewriters.

Baselines. For each task, we compare DocOpt against (1) *direct retrieval* without document transformation. In the visual setting, we use an OCR model as the direct retrieval baseline, as we focus only on text retrieval. (2) *zero-shot document transformations* without any training, via proprietary LLMs or the Qwen checkpoint we initiate DocOpt from. In open-weight retriever settings, we additionally evaluate (3) *retriever fine-tuning* and (4) *DocOpt - joint adaptation* of the retriever and transformation policy.

Evaluation. We use nDCG@5 as our evaluation metric. We use a small train split for finetuning and training (exact details are provided in Appendix A.5). The *direct retrieval* and *zero-shot transformation* baselines are applied to the test queries directly, and the *policy optimization*, *retriever fine-tuning* and *joint adaptation* methods are first trained on the train split and then evaluated similarly on the test set. To ensure a fair comparison, we allocate equal tuning effort to all three.

4.2 Results

Table 1 shows that document transformation improves retrieval quality across retrievers and datasets. Zero-shot rewriting already leads to retrieval gains over direct retrieval, with larger gains for Qwen-3

Table 1: Retrieval results across code and visual domains. Deltas are absolute changes relative to Direct Retrieval. Mean weights the two domains equally. Visual Direct Retrieval uses OCR. Bold marks the best displayed value per retriever and column.

Retriever	Method	Code		Visual		Mean	
		val	Δ	val	Δ	val	Δ
LEXICAL							
BM25	Direct Retrieval	15.23	0.0	14.13	0.0	14.68	0.0
	GPT-5.4-mini	37.90	+22.67	15.91	+1.78	26.90	+12.22
	Claude Haiku 4.5	46.37	+31.13	18.25	+4.13	32.31	+17.63
	Qwen-3 2B/4B	53.03	+37.80	18.05	+3.93	35.54	+20.86
	DocOpt	**58.20**	+42.97	**19.93**	+5.80	**39.06**	+24.38
CLOSED-WEIGHTS							
OpenAI text-emb-small	Direct Retrieval	68.63	0.0	48.08	0.0	58.35	0.0
	GPT-5.4-mini	70.84	+2.20	50.83	+2.75	60.83	+2.48
	Claude Haiku 4.5	71.94	+3.31	55.57	+7.49	63.75	+5.40
	Qwen-3 2B/4B	75.13	+6.50	53.33	+5.25	64.23	+5.88
	DocOpt	**78.23**	+9.60	**57.65**	+9.58	**67.94**	+9.59
OpenAI text-emb-large	Direct Retrieval	77.73	0.0	54.58	0.0	66.15	0.0
	GPT-5.4-mini	75.79	-1.94	55.81	+1.23	65.80	-0.35
	Claude Haiku 4.5	77.22	-0.52	58.00	+3.42	67.61	+1.45
	Qwen-3 2B/4B	78.70	+0.97	57.03	+2.45	67.86	+1.71
	DocOpt	**81.87**	+4.13	**59.40**	+4.83	**70.63**	+4.48
OPEN-WEIGHTS							
Qwen3-Emb-0.6B	Direct Retrieval	79.00	0.0	39.53	0.0	59.26	0.0
	GPT-5.4-mini	79.83	+0.83	46.92	+7.39	63.37	+4.11
	Claude Haiku 4.5	80.07	+1.07	37.34	-2.19	58.70	-0.56
	Qwen-3 2B/4B	80.70	+1.70	52.53	+13.00	66.61	+7.35
	DocOpt	81.93	+2.93	**57.35**	+17.83	**69.64**	+10.38
	Retriever Fine-tuning	80.87	+1.87	56.28	+16.75	68.57	+9.31
	DocOpt - Joint Adaptation	**82.50**	+3.50	54.33	+14.80	68.41	+9.15
Qwen3-Emb-4B	Direct Retrieval	83.43	0.0	44.50	0.0	63.97	0.0
	GPT-5.4-mini	81.31	-2.12	47.83	+3.33	64.57	+0.61
	Claude Haiku 4.5	81.67	-1.77	43.39	-1.11	62.53	-1.44
	Qwen-3 2B/4B	82.97	-0.47	56.25	+11.75	69.61	+5.64
	DocOpt	**84.63**	+1.20	57.70	+13.20	71.17	+7.20
	Retriever Fine-tuning	84.17	+0.73	**59.23**	+14.73	**71.70**	+7.73
	DocOpt - Joint Adaptation	84.20	+0.77	57.98	+13.48	71.09	+7.12
Jina-Colbert-V2	Direct Retrieval	58.87	0.0	53.75	0.0	56.31	0.0
	GPT-5.4-mini	68.67	+9.80	52.81	-0.94	60.74	+4.43
	Claude Haiku 4.5	71.02	+12.15	53.05	-0.71	62.03	+5.72
	Qwen-3 2B/4B	70.50	+11.63	55.83	+2.07	63.16	+6.85
	DocOpt	73.87	+15.00	58.03	+4.27	65.95	+9.64
	Retriever Fine-tuning	71.53	+12.67	56.75	+3.00	64.14	+7.83
	DocOpt - Joint Adaptation	**75.00**	+16.13	**63.30**	+9.55	**69.15**	+12.84

models over commerical API models. DocOpt in turn yields further gains, outperforming direct retrieval and zero-shot transformation on average across all retrievers (Appendix E details results per dataset in each domain). It demonstrates the benefit of not merely using an LLM to produce richer document representations, but to explicitly optimize those representations for the target retriever. For example, with BM25, DocOpt reaches 39.06 mean nDCG@5, compared with 26.90 for GPT-5.4-mini and 32.31 for Claude Haiku 4.5; with OpenAI text-embedding-3-small, it reaches 67.94, compared with 60.83 and 63.75, respectively.

Table 2: Estimated policy-preparation and rewriting time and cost at 10K and 1M documents, using a 150-step policy budget. DocOpt is cheapest at both scales and fastest at 1M. Totals are calculated before rounding.

Docs	Method	Policy preparation		Rewriting		Total	
		Time (h)	Cost ($)	Time (h)	Cost ($)	Time (h)	Cost ($)
10K	DocOpt (H100)	0.54	1.95	0.05	0.17	0.59	**2.12**
	GPT-5.4 mini	—	—	0.15	6.07	**0.15**	6.07
	Claude Haiku 4.5	—	—	0.26	9.74	0.26	9.74
	GPT-5.4	—	—	0.33	21.01	0.33	21.01
	Claude Sonnet 4.6	—	—	0.80	34.55	0.80	34.55
1M	DocOpt (H100)	0.54	1.95	4.77	16.64	**5.31**	**18.59**
	GPT-5.4 mini	—	—	15.24	607.49	15.24	607.49
	Claude Haiku 4.5	—	—	26.37	974.36	26.37	974.36
	GPT-5.4	—	—	33.06	2,101.32	33.06	2,101.32
	Claude Sonnet 4.6	—	—	79.90	3,454.62	79.90	3,454.62

In the black-box setting, DocOpt yields substantial gains for both OpenAI retrievers and across both domains. For text-embedding-3-small, it improves mean nDCG@5 from 58.35 to 67.94, corresponding to nearly identical gains on code (68.63→78.23) and VDR (48.08→57.65). Notably, the optimized small model even surpasses direct retrieval with the 6.5× more expensive text-embedding-3-large model (66.15 mean nDCG@5). DocOpt also improves the larger retriever itself, from 66.15 to 70.63 on average, showing that document optimization provides gains even when the underlying retriever is already strong.

In the open-weight setting, DocOpt remains competitive with directly fine-tuning the retriever, despite modifying only the document representations. It outperforms retriever fine-tuning for Qwen3-Embedding-0.6B (69.64 vs. 68.57) and Jina-ColBERT-V2 (65.95 vs. 64.14), while closely matching it for Qwen3-Embedding-4B (71.17 vs. 71.70). The gains are particularly large on VDR for the Qwen retrievers, where DocOpt improves Qwen3-Embedding-0.6B by 17.83 points and Qwen3-Embedding-4B by 13.20 points, and on code for Jina-ColBERT-V2, where it improves performance by 15.00 points. Moreover, document-side and retriever-side adaptation can be complementary: jointly optimizing both yields the strongest result for Jina-ColBERT-V2, reaching 69.15 mean nDCG@5 compared with 65.95 for DocOpt alone and 64.14 for retriever fine-tuning alone.

5 Analysis

End-to-End Costs. We use vLLM (Kwon et al., 2023) for corpus rewriting and Hugging Face TRL for GRPO training. Our cost comparison uses Qwen3-4B-Instruct-2507 on DS1000Retrieval (Lai et al., 2022), whose documents average 185 source tokens. We time rewriting over all 1,998 documents per method and extrapolate to 10K and 1M documents.

On our hardware, representative text-policy runs averaged 13.87/10.17 seconds per step on a single A100/H100. Training uses 407 queries against a fixed 1,998-document index; the 150-step budget includes full-index refreshes after steps 50 and 100. We reuse the same policy at both deployment scales. This gives 0.54 H100-hours, including approximately 6.9 minutes of initialization, refresh, and other overhead. Preparation costs $1.95, including $0.07 for retriever embeddings.

API costs use official Batch pricing. DocOpt GPU time is valued at $3.49/H100-hour. Target-index construction and unrecorded query-generation time are excluded. As Table 2 shows, DocOpt costs $2.12 at 10K documents, below all API baselines. At 1M documents, its preparation and rewriting cost approximately 33× less and run 2.9× faster than GPT-5.4 mini ($18.59 vs. $607.49; 5.31 vs. 15.24 hours).

Qualitative Analysis. We conduct qualitative analysis using Qwen3-Embedding-4B as the retriever on the DS10K dataset. The baseline retrieval performance on the original dataset is 61.2 nDCG@5, which remains largely unchanged after zero-shot transformation (61.1), but improves to 63.8 following document optimization. We analyze the distribution of document lengths across original, zero-shot

Table 3: nDCG@5 on BRIGHT using Qwen3-Embedding-0.6B and OpenAI text-embedding-3-small. DocOpt achieves the best average retrieval performance with both retrievers, with larger gains for Qwen3-Embedding-0.6B.

Method	Avg.	Bio.	Earth	Econ.	Psych.	Robot.	Sust.	SO
Qwen3-Embedding-0.6B								
Direct Retrieval	0.1306	0.1063	0.1992	0.1528	0.1431	0.1128	0.0998	**0.0999**
Zero-shot Transformation	0.1272	0.1170	0.2378	0.1458	0.1480	0.0941	0.0872	0.0608
DocOpt	**0.1463**	**0.1188**	**0.2505**	**0.1820**	**0.1532**	**0.1173**	**0.1114**	0.0908
OpenAI text-embedding-3-small								
Direct Retrieval	0.1647	**0.2010**	0.2701	**0.1718**	**0.2200**	**0.0995**	0.1298	0.0606
Zero-shot Transformation	0.1545	0.1688	0.2994	0.1578	0.1912	0.0842	0.1252	0.0549
DocOpt	**0.1670**	0.1832	**0.3174**	0.1670	0.1787	0.0938	**0.1412**	**0.0876**

Table 4: Transfer across retrievers on ViDoRe2. DocOpt transfers most effectively within retriever families, while cross-family transfer is less consistent.

Rewrite Policy	Qwen 0.6B	Qwen 4B	OpenAI small	OpenAI large
No training (Baseline)	52.53	56.25	53.33	57.03
Qwen-trained	**57.35**	**57.65**	52.19	57.74
OpenAI-trained	51.52	55.95	**57.65**	**58.56**

transformed, and optimized documents and provide the statistics here (also visualized in Appendix C). Original documents have a mean of 183 tokens and a median of 143. Zero-shot transformations substantially increase length, with a mean of 382 and a median of 355, nearly doubling document size. In contrast, optimized documents are more compact than zero-shot rewrites, with a mean of 346 and a median of 311. This indicates that the learned policy does not simply expand content, but selectively refines it to retain useful information, consistent with the goal of our approach. Appendix C additionally includes a representation space analysis of queries, original documents, optimized documents, and negatives using a two-dimensional t-SNE projection for qualitative inspection.

Large Scale Natural Language Retrieval. We additionally test DocOpt on a large-scale natural language retrieval setting. We use the StackExchange subset of the BRIGHT benchmark (SU et al., 2025) for evaluation, which contains 7 datasets spanning diverse domains with over 500K documents in total. Table 3 shows that DocOpt remains effective across both retrievers. With Qwen3-Embedding-0.6B, DocOpt improves average nDCG@5 by 12.0% over direct retrieval and 15.0% over zero-shot transformation, outperforming direct retrieval on 6 of 7 domains and zero-shot transformation on all 7. With OpenAI text-embedding-3-small, gains are more modest: DocOpt improves the average by 1.4% over direct retrieval and 8.1% over zero-shot transformation, outperforming them on 3 and 6 of 7 domains, respectively.

Transfer across Retrievers. We test whether document transformations learned for one retriever transfer to another retriever. We run DocOpt against Qwen3-Embedding-0.6B and against OpenAI text-embedding-3-small, and test how well the rewriting policies generalize to other retrievers. Table 4 illustrates that DocOpt transfers more effectively within retriever families. Qwen-trained rewrites improve Qwen 4B by 1.40 nDCG@5 points, while OpenAI-trained rewrites improve OpenAI large by 1.53 points. Across-family transfer is less consistent: Qwen-trained rewrites improve OpenAI large by 0.71 points, whereas it reduces OpenAI-small by 1.14 and OpenAI-trained rewrites reduce Qwen 4B performance by 0.30 points.

Training on Synthetic queries. Table 5 shows synthetic-query supervision improves average retrieval over zero-shot rewriting for both retrievers, while remaining below labeled-query supervision. Gains vary across collections and are larger for OpenAI small, with slight declines on ESG for both retrievers. Appendix B provides the implementation details.

Table 5: Learning from synthetic queries on ViDoRe2. Synthetic-query supervision improves average retrieval over zero-shot rewriting for both retrievers, but remains below labeled-query supervision.

Supervision	Bio.	Econ.	ESG	ESG-H	Mean
Qwen3-Embedding-0.6B					
Zero-shot	54.40	45.80	55.80	54.10	52.53
Labeled queries	58.50	53.90	58.80	58.20	57.35
Synthetic queries	55.70	46.10	55.30	54.10	52.80
OpenAI text-embedding-3-small					
Zero-shot	53.80	47.00	56.90	55.60	53.33
Labeled queries	55.50	51.40	61.80	61.90	57.65
Synthetic queries	54.60	48.40	56.30	57.70	54.25

Appendix D contains ablations of key design choices in our method, focusing on the reward formulation and negative query mining strategy.

6 Related Work

Document-side adaptation. Document expansion (Berger & Lafferty, 1999; Tao et al., 2006; Pickens et al., 2010; Efron et al., 2012) is a longstanding technique in information retrieval, designed to address the vocabulary mismatch problem (Furnas et al., 1987) in lexical retrieval. Later expansion solutions leverage neural models to predict likely queries and augment documents accordingly (Nogueira et al., 2019; Nogueira, 2019; Gospodinov et al., 2023; Kuo et al., 2025). None of these treat document adaptation as an optimization problem, nor do they directly optimize it for retrieval performance. Closest to our work, Liu et al. (2025a) jointly optimize query and document rewriting with reinforcement learning, and find only modest gains from document-only optimization. However, their document-only conclusion is based on NFCorpus, where queries are extremely short (median length of two words), which may underrepresent settings with more nuanced information needs. In contrast, we focus on document-side optimization that incurs no additional query-time cost, across more diverse retrieval settings.

Query-side methods and multi-stage retrieval. Many lines of work improve retrieval by modifying the query or introducing additional stages at inference time. Query expansion (Voorhees, 1994; Berger & Lafferty, 1999; Xu & Croft, 2000) and pseudo-relevance feedback (Yu et al., 2021; Sung et al., 2023) augment the input query with additional terms, while more recent approaches leverage language models for query rewriting (Gao et al., 2023; Hsu et al., 2024). In parallel, multi-stage retrieval pipelines (Kuzi et al., 2020; Chen et al., 2022; Uzan et al., 2025) combine an initial retrieval step with reranking models to refine results. While these methods can substantially improve retrieval quality, they incur additional query-time cost and system complexity.

Visual Document Retrieval. Existing approaches can be broadly divided into image-based and text-based methods. Single-vector image-based models embed page images and queries into a shared embedding space (Radford et al., 2021), while more recent architectures adopt multi-vector representations to capture fine-grained interactions (Faysse et al., 2025). In contrast, text-based approaches convert documents into textual surrogates using OCR or vision-language models (Berrios et al., 2023; Nguyen et al., 2025), and leverage the mature infrastructure and ecosystem of text retrieval. We focus on this setting; however, unlike prior work that treats the surrogate heuristically, we optimize it to better align with the target retriever.

7 Conclusion

We presented document optimization as a framework for improving retrieval by rewriting documents to better align with a target retriever. Our method requires only black-box access, applies across retrieval architectures, and improves performance on both visual and code retrieval tasks. These results suggest that optimizing the document space itself is a promising and underexplored direction. We encourage future work to extend the framework to other domains and tasks, and further study the interplay between document-side optimization and model-side adaptation.

DISCLOSURE OF LLM USE

Generative AI tools were used to assist with manuscript editing and wording, the production and refinement of some plots and diagrams, implementation assistance, and experimental design. All AI-assisted outputs were critically reviewed and verified by the authors, who take full responsibility for the content of the paper.

REFERENCES

Anthropic. Introducing claude haiku 4.5, October 2025. URL https://www.anthropic.com/news/claude-haiku-4-5.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021. URL https://arxiv.org/abs/2108.07732.

Adam Berger and John Lafferty. Information retrieval as statistical translation. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, pp. 222–229, New York, NY, USA, 1999. Association for Computing Machinery. ISBN 1581130961. doi: 10.1145/312624.312681. URL https://doi.org/10.1145/312624.312681.

William Berrios, Gautam Mittal, Tristan Thrush, Douwe Kiela, and Amanpreet Singh. Towards language models that can see: Computer vision through the lens of natural language, 2023. URL https://arxiv.org/abs/2306.16410.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. 2021.

Tao Chen, Mingyang Zhang, Jing Lu, Michael Bendersky, and Marc Najork. Out-of-domain semantics to the rescue! Zero-Shot hybrid retrieval models. In *Advances in Information Retrieval: 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10–14, 2022, Proceedings, Part I*, pp. 95–110, Berlin, Heidelberg, 2022. Springer-Verlag. ISBN 978-3-030-99735-9. doi: 10.1007/978-3-030-99736-6_7. URL https://doi.org/10.1007/978-3-030-99736-6_7.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding, 2019. URL https://arxiv.org/abs/1810.04805.

Miles Efron, Peter Organisciak, and Katrina Fenlon. Improving retrieval of short texts through document expansion. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, pp. 911–920, New York, NY, USA, 2012. Association for Computing Machinery. ISBN 9781450314725. doi: 10.1145/2348283.2348405. URL https://doi.org/10.1145/2348283.2348405.

Manuel Faysse, Hugues Sibille, Tony Wu, Bilel Omrani, Gautier Viaud, Celine Hudelot, and Pierre Colombo. ColPali: Efficient document retrieval with vision language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=ogjBpZ8uSi.

G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. The vocabulary problem in human-system communication. *Commun. ACM*, 30(11):964–971, November 1987. ISSN 0001-0782. doi: 10.1145/32206.32212. URL https://doi.org/10.1145/32206.32212.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. Precise zero-shot dense retrieval without relevance labels. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1762–1777, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.99. URL https://aclanthology.org/2023.acl-long.99/.

Mitko Gospodinov, Sean MacAvaney, and Craig Macdonald. Doc2query–: When less is more. In *Advances in Information Retrieval: 45th European Conference on Information Retrieval, ECIR 2023, Dublin, Ireland, April 2–6, 2023, Proceedings, Part II*, pp. 414–422, Berlin, Heidelberg, 2023. Springer-Verlag. ISBN 978-3-031-28237-9. doi: 10.1007/978-3-031-28238-6_31. URL https://doi.org/10.1007/978-3-031-28238-6_31.

Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In Yee Whye Teh and Mike Titterington (eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 297–304, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/gutmann10a.html.

Sheryl Hsu, Omar Khattab, Chelsea Finn, and Archit Sharma. Grounding by trying: LLMs with reinforcement learning-enhanced retrieval, 2024. URL https://arxiv.org/abs/2410.23214.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus, 2017. URL https://arxiv.org/abs/1702.08734.

Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011. doi: 10.1109/TPAMI.2010.57.

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/.

Omar Khattab and Matei Zaharia. 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*, SIGIR '20, pp. 39–48, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450380164. doi: 10.1145/3397271.3401075. URL https://doi.org/10.1145/3397271.3401075.

Tzu-Lin Kuo, Wei-Ning Chiu, Wei-Yun Ma, and Pu-Jen Cheng. Doc2query++: Topic-coverage based document expansion and its application to dense retrieval via dual-index fusion, 2025. URL https://arxiv.org/abs/2510.09557.

Saar Kuzi, Mingyang Zhang, Cheng Li, Michael Bendersky, and Marc Najork. Leveraging semantic and lexical matching to improve the recall of document retrieval systems: A hybrid approach, 2020. URL https://arxiv.org/abs/2010.01195.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention, 2023. URL https://arxiv.org/abs/2309.06180.

Yuhang Lai, Chengxi Li, Yiming Wang, Tianyi Zhang, Ruiqi Zhong, Luke Zettlemoyer, Scott Wen tau Yih, Daniel Fried, Sida Wang, and Tao Yu. Ds-1000: A natural and reliable benchmark for data science code generation, 2022. URL https://arxiv.org/abs/2211.11501.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 6086–6096, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1612. URL https://aclanthology.org/P19-1612/.

Lei Li, Yuqi Wang, Runxin Xu, Peiyi Wang, Xiachong Feng, Lingpeng Kong, and Qi Liu. Multimodal ArXiv: A dataset for improving scientific comprehension of large vision-language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14369–14387, Bangkok, Thailand, August 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.acl-long.775.

Jingming Liu, Yumeng Li, Wei Shi, Yao-Xiang Ding, Hui Su, and Kun Zhou. Harnessing the power of reinforcement learning for language-model-based information retriever via query-document co-augmentation, 2025a. URL https://openreview.net/forum?id=6UDRtcScwO.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective, 2025b. URL https://arxiv.org/abs/2503.20783.

Quentin Macé, António Loison, and Manuel Faysse. ViDoRe benchmark v2: Raising the bar for visual retrieval. *arXiv:2505.17166*, 2025. URL https://arxiv.org/abs/2505.17166.

Yu A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 42 (4):824–836, April 2020. ISSN 0162-8828. doi: 10.1109/TPAMI.2018.2889473. URL https://doi.org/10.1109/TPAMI.2018.2889473.

Minesh Mathew, Viraj Bagal, Rubèn Pérez Tito, Dimosthenis Karatzas, Ernest Valveny, and C. V Jawahar. InfographicVQA, 2021a. URL https://arxiv.org/abs/2104.12756.

Minesh Mathew, Dimosthenis Karatzas, and C. V. Jawahar. DocVQA: A dataset for VQA on document images, 2021b. URL https://arxiv.org/abs/2007.00398.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. MTEB: Massive text embedding benchmark, 2023. URL https://arxiv.org/abs/2210.07316.

Thong Nguyen, Yibin Lei, Jia-Huei Ju, and Andrew Yates. SERVAL: Surprisingly effective zero-shot visual document retrieval powered by large vision and language models. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 30807–30822, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1568. URL https://aclanthology.org/2025.emnlp-main.1568/.

Rodrigo Nogueira. From doc2query to doctttttquery. 2019. URL https://api.semanticscholar.org/CorpusID:209423937.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. Document expansion by query prediction, 2019. URL https://arxiv.org/abs/1904.08375.

OpenAI. Introducing gpt-5.4 mini and nano, March 2026. URL https://openai.com/index/introducing-gpt-5-4-mini-and-nano/.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

Jeremy Pickens, Matthew Cooper, and Gene Golovchinsky. Reverted indexing for feedback and expansion. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pp. 1049–1058, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781450300995. doi: 10.1145/1871437.1871571. URL https://doi.org/10.1145/1871437.1871571.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, Xuanhui Wang, and Michael Bendersky. Large language models are effective text rankers with pairwise ranking prompting. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 1504–1518, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-naacl.97. URL https://aclanthology.org/2024.findings-naacl.97/.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. URL https://arxiv.org/abs/2103.00020.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese bert-networks, 2019. URL https://arxiv.org/abs/1908.10084.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389, April 2009. ISSN 1554-0669. doi: 10.1561/1500000019. URL https://doi.org/10.1561/1500000019.

Keshav Santhanam, Omar Khattab, Christopher Potts, and Matei Zaharia. Plaid: An efficient engine for late interaction retrieval, 2022a. URL https://arxiv.org/abs/2205.09707.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz (eds.), *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3715–3734, Seattle, United States, July 2022b. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.272. URL https://aclanthology.org/2022.naacl-main.272/.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Hongjin SU, Howard Yen, Mengzhou Xia, Weijia Shi, Niklas Muennighoff, Han yu Wang, Liu Haisu, Quan Shi, Zachary S Siegel, Michael Tang, Ruoxi Sun, Jinsung Yoon, Sercan O Arik, Danqi Chen, and Tao Yu. BRIGHT: A realistic and challenging benchmark for reasoning-intensive retrieval. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=ykuc5q381b.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. Is ChatGPT good at search? investigating large language models as re-ranking agents. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 14918–14937, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.923. URL https://aclanthology.org/2023.emnlp-main.923/.

Mujeen Sung, Jungsoo Park, Jaewoo Kang, Danqi Chen, and Jinhyuk Lee. Optimizing test-time query representations for dense retrieval. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki

(eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 5731–5746, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.354. URL https://aclanthology.org/2023.findings-acl.354/.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In S. Solla, T. Leen, and K. Müller (eds.), *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999. URL https://proceedings.neurips.cc/paper_files/paper/1999/file/464d828b85b0bed98e80ade0a5c43b0f-Paper.pdf.

Tao Tao, Xuanhui Wang, Qiaozhu Mei, and ChengXiang Zhai. Language model information retrieval with document expansion. In Robert C. Moore, Jeff Bilmes, Jennifer Chu-Carroll, and Mark Sanderson (eds.), *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pp. 407–414, New York City, USA, June 2006. Association for Computational Linguistics. URL https://aclanthology.org/N06-1052/.

Omri Uzan, Asaf Yehudai, Roi pony, Eyal Shnarch, and Ariel Gera. Guided query refinement: Multimodal hybrid retrieval with test-time optimization, 2025. URL https://arxiv.org/abs/2510.05038.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding, 2019. URL https://arxiv.org/abs/1807.03748.

Ellen M. Voorhees. Query expansion using lexical-semantic relations. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '94, pp. 61–69, Berlin, Heidelberg, 1994. Springer-Verlag. ISBN 038719889X.

Orion Weller, Kyle Lo, David Wadden, Dawn Lawrie, Benjamin Van Durme, Arman Cohan, and Luca Soldaini. When do generative query and document expansions fail? a comprehensive study across methods, retrievers, and datasets. In Yvette Graham and Matthew Purver (eds.), *Findings of the Association for Computational Linguistics: EACL 2024*, pp. 1987–2003, St. Julian's, Malta, March 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-eacl.134. URL https://aclanthology.org/2024.findings-eacl.134/.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3–4):229–256, May 1992. ISSN 0885-6125. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

Jinxi Xu and W. Bruce Croft. Improving the effectiveness of information retrieval with local context analysis. *ACM Trans. Inf. Syst.*, 18(1):79–112, January 2000. ISSN 1046-8188. doi: 10.1145/333135.333138. URL https://doi.org/10.1145/333135.333138.

HongChien Yu, Chenyan Xiong, and Jamie Callan. Improving query representations for dense retrieval with pseudo relevance feedback. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, pp. 3592–3596, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450384469. doi: 10.1145/3459637.3482124. URL https://doi.org/10.1145/3459637.3482124.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Fengbin Zhu, Wenqiang Lei, Fuli Feng, Chao Wang, Haozhou Zhang, and Tat-Seng Chua. Towards complex document understanding by discrete reasoning. In *Proceedings of the 30th ACM International Conference on Multimedia*, pp. 4857–4866, 2022.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences, 2020. URL https://arxiv.org/abs/1909.08593.

A Appendix

A.1 Retrieval Metrics

Let $y_i \in \{0, 1\}$ denote the relevance of the document at rank i.

Recall@k.

$$\text{Recall@}k = \frac{\sum_{i=1}^{k} y_i}{\sum_{i=1}^{n} y_i}. \tag{5}$$

nDCG@k.

$$\text{nDCG@}k = \frac{\sum_{i=1}^{k} \frac{y_i}{\log_2(i+1)}}{\max_\pi \sum_{i=1}^{k} \frac{y_{\pi(i)}}{\log_2(i+1)}}, \tag{6}$$

where the denominator corresponds to the ideal (best possible) ranking.

A.2 Document Transformation Prompts

Visual Document Transformation Prompt

```
Provide a comprehensive description of the document in the image in English.
Begin with a summary, then follow with details. Extract all visible text and numerica
values from the document.
```

Also used in Nguyen et al. (2025)

Code Transformation Prompt

```
Improve this code snippet.

You may:
- Add a short natural-language summary.
- Add comments where helpful.
- Rename variables or refactor for readability.

STRICT RULES:
- DO NOT change the behavior or functionality of the source document.
- Preserve the exact input-output semantics.
- Do not add facts, assumptions, or code behavior that is not supported by the source
- Keep the output concise and high-signal.

Return only the updated code with comments/documentation.
Do not add text outside the code.
```

A.3 GRPO Configuration

We use the TRL library for optimization. We train in BF16 precision, without LoRA. We run each training run for 150 steps, and cap the rollouts at 1024 tokens. We use these hyperparameters:

- `index_refresh_rate` = 50
- `batch_size` = 32
- `num_generations` = 8
- `learning_rate` = 1×10^{-5}
- `beta` = 0.1
- `max_document_tokens` = 1024,
- `max_prompt_length` = 4096,

- `temperature` = 0.7,
- `top_p` = 0.8,
- `top_k` = 20,
- `repetition_penalty` = 1.0,
- `presence_penalty` = 0

A.4 MODELS

We use a diverse set of retrievers and language models to evaluate our approach across architectures and access settings. Table 6 lists lexical and neural retrievers, including both open- and closed-weight models, as well as single- and multi-vector variants. Table 7 summarizes the instruction-tuned language models used to parameterize the transformation policy, covering both text-only and vision-language settings.

Retriever	**Weights**	**Params**	**Dim**	**Vector Type**
`BM25`	Open, non-neural baseline	N/A	N/A	
`text-embedding-3-small`	Closed	Undisclosed	1536	Single
`text-embedding-3-large`	Closed	Undisclosed	3072	Single
`Qwen3-Embedding-0.6B`	Open	0.6B	1024	Single
`Qwen3-Embedding-4B`	Open	4B	2560	Single
`jina-colbert-v2`	Open	0.56B	128	Multi

Table 6: Retrievers used in our experiments.

Language Model	Closed/Open	Param Size
`Qwen/Qwen3-4B-Instruct-2507`	Open	4B
`Qwen/Qwen3-VL-2B-Instruct`	Open	2B
GPT-5.4-mini	Closed	Undisclosed
Claude Haiku 4.5	Closed	Undisclosed
GPT-5.4	Closed	Undisclosed
Claude Sonnet 4.6	Closed	Undisclosed

Table 7: Language models used in our experiments. Parameter counts for closed-weight models are not publicly disclosed.

A.5 EVALUATION

We evaluate on both code and visual document retrieval datasets. We assume an in-domain supervised setting. We randomly sample 20% of the benchmark queries for finetuning/training and maintain consistent splits across all experiments. Importantly, this split provides limited direct supervision relative to what is evaluated. Only 21% of overall evaluated documents are relevant to any training query, and only 4% of held-out queries have a relevant document that also received supervision during training. Because the policy is applied to the entire collection after training, the large majority of evaluated documents were rewritten by a policy that never received reward signal for them. Test performance therefore primarily reflects the policy's ability to generalize a learned rewriting strategy to unsupervised documents and held-out queries, rather than memorization of training targets. Following standard practice in retriever training, for each document we use its associated positive queries as Q_i^+ and construct Q_i^- by mining 5 hard negative queries from the positive queries associated with other documents in the training split. We use this setting throughout the main experiments; Appendix D compares it against random negative sampling and other numbers of negatives. We train a separate policy for each dataset and retriever, initialized from the same instruction-tuned checkpoint. We evaluate the final policy checkpoint after 150 optimization steps. We use a fixed set of policy-training hyperparameters across datasets and retrievers, without test

queries leakage. The code datasets (Table 8) are drawn from MTEB and provide diverse programming tasks with natural language queries. The visual datasets (Table 9), from Vidore2, consist of document collections across multiple domains, enabling evaluation in multimodal retrieval settings.

Dataset	Num Docs	Num Queries
`HumanEvalRetrieval`	158	127 test / 31 train
`MBPPRetrieval`	974	779 test / 195 train
`DS1000Retrieval`	1998	1591 test / 407 train

Table 8: Code datasets (MTEB) used in our experiments.

Dataset	Num Docs	Num Queries
`biomedical_lectures_v2`	1016	513 test / 128 train
`economics_reports_v2`	452	186 test / 46 train
`esg_reports_v2`	1538	183 test / 45 train
`esg_reports_human_labeled_v2`	1538	42 test / 10 train

Table 9: Visual datasets (Vidore2) used in our experiments.

Dataset	Num Docs	Num Queries
`biology`	57,359	83 test / 20 train
`earth_science`	121,249	93 test / 23 train
`economics`	50,220	83 test / 20 train
`psychology`	52,835	81 test / 20 train
`robotics`	61,961	81 test / 20 train
`sustainable_living`	60,792	87 test / 21 train
`stackoverflow`	107,081	94 test / 23 train

Table 10: StackExchange datasets from BRIGHT used in our experiments.

A.5.1 Hyperparameters: Retrieval, and vLLM

Unless otherwise stated, we use the same hyperparameters across MTEB and Vidore2 runs.

vLLM Generation

- `vllm_batch_size` = 128
- `max_queries_tokens` = 8192
- `max_document_tokens` = 4096
- `temperature` = 0.3
- `top_p` = 0.9
- `top_k` = 30
- `repetition_penalty` = 1.1
- `presence_penalty` = 0.0

Retrieval and Rewards

- ColBERT settings: `colbert_doc_maxlen` = 524, `colbert_query_maxlen` = 64
- `retriever_api_batch_size` = 64
- `retriever_batch_size` = 32

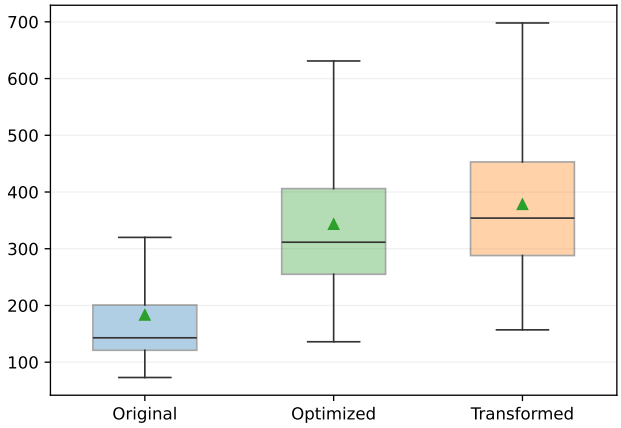

Figure 3: Distribution of document lengths (in tokens) for original, optimized documents and (zero-shot) transformed, on the DS1000 dataset using Qwen3-Embedding-4B as the retriever.

Type	Setting	HumanEval	MBPP	DS10K	AVG
Zero-shot Transformation		93.6	84.0	47.8	75.1
Similarity	Positives Only	94.5	**84.8**	49.1	76.1
	Positives + Negatives	93.1	**84.8**	51.3	76.4
Ranking	Positives Only	92.9	84.2	50.9	76.0
	Positives + Negatives	**94.9**	**84.8**	**52.1**	**77.3**

Table 11: Comparison of reward function variants.

B Synthetic Queries

We use the frozen Qwen3-VL-2B-Instruct model to generate English search queries from page images, without exposing it to benchmark queries or relevance labels. We restrict generation to pages associated with the labeled training split and subsample the synthetic queries to match its query budget and approximate its per-page distribution. Each source page is treated as relevant, and hard-negative queries are mined exclusively from the synthetic bank. We retain the same retrieval corpus, training configuration, and 150-step optimization budget, and evaluate the final policy on the same held-out human queries.

C Additional Plots

Figure 4 visualizes embeddings of queries, original documents, optimized documents, and negatives using a two-dimensional t-SNE projection for qualitative inspection (additional examples are provided in Figure 5 in the appendix). While 2D distances do not faithfully reflect similarities in the original space, they provide a useful approximation of the local geometry. We observe that optimized documents form clusters and in many cases where optimization improves nDCG@5, shift closer to the query and farther from negatives. Importantly, improvements are not solely driven by reduced query–document distance, but by favorable relative positioning with respect to competing negatives, consistent with the ranking-based reward.

D Ablations and Design Choices

We analyze key design choices in our approach, focusing on the reward formulation and negative query construction. All ablations are conducted on three code retrieval datasets with OpenAI text-embedding-3-small as the retriever, using a reduced optimization budget but otherwise the same

Setting	#Queries	HumanEval	MBPP	DS10K	AVG
Zero-shot Transformation		93.6	84.0	47.8	75.1
Random Sampling	1	94.3	84.4	51.5	76.7
Hard Negatives	1	94.7	84.1	52.2	77.0
Random Sampling	5	92.9	83.9	51.4	76.1
Hard Negatives	5	94.9	**84.8**	**52.1**	**77.3**
Random Sampling	10	**95.1**	83.3	51.3	76.6
Hard Negatives	10	94.9	**84.8**	51.7	77.1

Table 12: Comparison of weak supervision strategies for negative queries.



Figure 4: Two-dimensional t-SNE projections of Qwen3-Embedding-4B embeddings across different queries.

training protocol as in the main experiments. Design choices were made on a separate validation split drawn from the training queries; for comparability with the main results, the tables report the same configurations evaluated on the test queries, which were not used for any design decisions.

Reward function choice. Table 11 shows that all learned reward variants improve over the zero-shot baseline, supporting document optimization with reinforcement learning as a general framework. The strongest overall results are achieved by the ranking-based reward that uses both positive and negative queries. Negative queries consistently help under both ranking-based and similarity-based objectives, indicating that discouraging spurious irrelevant matches is important for learning effective document transformations.

Negative Queries. Table 12 compares strategies for constructing negative queries, including random sampling and hard negative mining, as well as varying the number of negatives per document. We find that hard negatives outperform random sampling. Increasing the number of negatives yields diminishing returns, with the strongest results obtained using 5 hard negatives.

E Per-Domain Results

Retriever	Method	Human Eval		MBPP		DS10K		AVG	
		val	Δ	val	Δ	val	Δ	val	Δ
LEXICAL									
BM25	Direct Retrieval	20.4	0.0	2.4	0.0	22.9	0.0	15.23	0.0
	GPT-5.4-mini	51.4	+31.0	33.4	+31.0	28.8	+5.9	37.90	+22.67
	Claude Haiku 4.5	64.7	+44.3	42.0	+39.6	32.4	+9.5	46.37	+31.13
	Qwen3-4B-Instruct-2507	72.1	+51.7	51.2	+48.8	35.8	+12.9	53.03	+37.80
	DocOpt	**80.5**	+60.1	**57.7**	+55.3	**36.4**	+13.5	**58.20**	+42.97
CLOSED-WEIGHTS									
OpenAI text-emb-small	Direct Retrieval	84.1	0.0	74.6	0.0	47.2	0.0	68.63	0.0
	GPT-5.4-mini	92.8	+8.7	77.3	+2.7	42.5	-4.7	70.84	+2.20
	Claude Haiku 4.5	91.1	+7.0	79.9	+5.3	44.8	-2.4	71.94	+3.31
	Qwen3-4B-Instruct-2507	93.6	+9.5	84.0	+9.4	47.8	+0.6	75.13	+6.50
	DocOpt	**94.6**	+10.5	**85.2**	+10.6	**54.9**	+7.7	**78.23**	+9.60
OpenAI text-emb-large	Direct Retrieval	93.0	0.0	84.8	0.0	55.4	0.0	77.73	0.0
	GPT-5.4-mini	96.4	+3.4	83.9	-0.9	47.1	-8.3	75.79	-1.94
	Claude Haiku 4.5	**96.9**	+3.9	84.6	-0.2	50.1	-5.3	77.22	-0.52
	Qwen3-4B-Instruct-2507	94.8	+1.8	**87.2**	+2.4	54.1	-1.3	78.70	+0.97
	DocOpt	95.1	+2.1	**87.2**	+2.4	**63.3**	+7.9	**81.87**	+4.13
OPEN-WEIGHTS									
Qwen3-Emb-0.6B	Direct Retrieval	93.7	0.0	87.1	0.0	56.2	0.0	79.00	0.0
	GPT-5.4-mini	**98.4**	+4.7	88.6	+1.5	52.5	-3.7	79.83	+0.83
	Claude Haiku 4.5	97.7	+4.0	89.7	+2.6	52.8	-3.4	80.07	+1.07
	Qwen3-4B-Instruct-2507	96.3	+2.6	89.9	+2.8	55.9	-0.3	80.70	+1.70
	DocOpt	97.6	+3.9	**90.2**	+3.1	58.0	+1.8	81.93	+2.93
	Retriever Fine-tuning	96.1	+2.4	87.6	+0.5	58.9	+2.7	80.87	+1.87
	DocOpt - Joint Adaptation	98.1	+4.4	89.3	+2.2	**60.1**	+3.9	**82.50**	+3.50
Qwen3-Emb-4B	Direct Retrieval	98.1	0.0	91.0	0.0	61.2	0.0	83.43	0.0
	GPT-5.4-mini	96.1	-2.0	88.1	-2.9	59.7	-1.5	81.31	-2.12
	Claude Haiku 4.5	96.0	-2.1	88.4	-2.6	60.6	-0.6	81.67	-1.77
	Qwen3-4B-Instruct-2507	97.3	-0.8	90.5	-0.5	61.1	-0.1	82.97	-0.47
	DocOpt	**98.5**	+0.4	91.6	+0.6	**63.8**	+2.6	**84.63**	+1.20
	Retriever Fine-tuning	98.3	+0.2	**92.0**	+1.0	62.2	+1.0	84.17	+0.73
	DocOpt - Joint Adaptation	97.9	-0.2	91.1	+0.1	63.6	+2.4	84.20	+0.77
Jina-Colbert-V2	Direct Retrieval	64.3	0.0	76.2	0.0	36.1	0.0	58.87	0.0
	GPT-5.4-mini	83.7	+19.4	82.9	+6.7	39.5	+3.4	68.67	+9.80
	Claude Haiku 4.5	86.0	+21.7	86.3	+10.1	40.8	+4.7	71.02	+12.15
	Qwen3-4B-Instruct-2507	87.9	+23.6	85.2	+9.0	38.4	+2.3	70.50	+11.63
	DocOpt	91.7	+27.4	87.2	+11.0	42.7	+6.6	73.87	+15.00
	Retriever Fine-tuning	87.2	+22.9	86.4	+10.2	41.0	+4.9	71.53	+12.67
	DocOpt - Joint Adaptation	**92.9**	+28.6	**88.4**	+12.2	**43.7**	+7.6	**75.00**	+16.13

Table 13: Code document retrieval results. Deltas are absolute changes relative to Direct Retrieval. Bold marks the best displayed value per retriever and column.

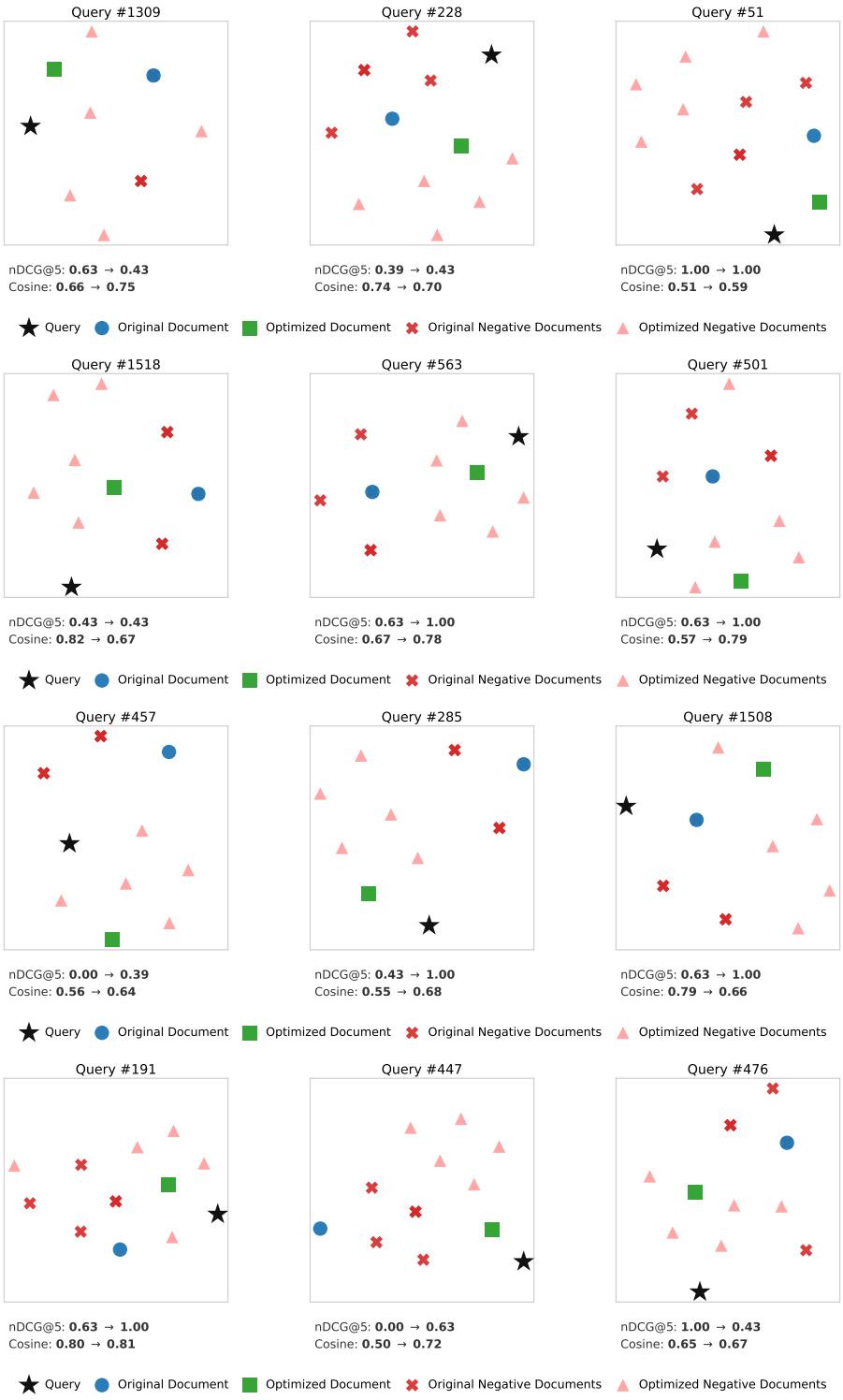

Figure 5: Two-dimensional t-SNE projections of Qwen3-Embedding-4B embeddings across representative queries. Each plot shows a single query with its original and optimized documents, along with corresponding negatives.

Retriever	Method	Biomed		Econ		ESG Human		ESG Full		AVG	
		val	Δ	val	Δ	val	Δ	val	Δ	val	Δ
LEXICAL											
BM25	Direct Retrieval	12.3	0.0	13.7	0.0	19.3	0.0	11.2	0.0	14.13	0.0
	GPT-5.4-mini	15.3	+3.0	15.6	+1.9	15.2	-4.1	**17.5**	+6.3	15.91	+1.78
	Claude Haiku 4.5	17.1	+4.8	**16.2**	+2.5	23.0	+3.7	16.7	+5.5	18.25	+4.13
	Qwen3-VL-2B-Instruct	16.7	+4.4	15.3	+1.6	28.1	+8.8	12.1	+0.9	18.05	+3.93
	DocOpt	**17.2**	+4.9	16.1	+2.4	**31.5**	+12.2	14.9	+3.7	**19.93**	+5.80
CLOSED-WEIGHTS											
OpenAI text-emb-small	Direct Retrieval	48.8	0.0	47.2	0.0	47.4	0.0	48.9	0.0	48.08	0.0
	GPT-5.4-mini	54.3	+5.5	46.4	-0.8	46.6	-0.8	56.0	+7.1	50.83	+2.75
	Claude Haiku 4.5	**56.3**	+7.5	48.3	+1.1	59.2	+11.8	58.5	+9.6	55.57	+7.49
	Qwen3-VL-2B-Instruct	53.8	+5.0	47.0	-0.2	55.6	+8.2	56.9	+8.0	53.33	+5.25
	DocOpt	55.5	+6.7	**51.4**	+4.2	**61.9**	+14.5	**61.8**	+12.9	**57.65**	+9.58
OpenAI text-emb-large	Direct Retrieval	54.3	0.0	52.8	0.0	56.8	0.0	54.4	0.0	54.58	0.0
	GPT-5.4-mini	**60.3**	+6.0	48.5	-4.3	57.1	+0.3	57.3	+2.9	55.81	+1.23
	Claude Haiku 4.5	59.4	+5.1	52.6	-0.2	61.5	+4.7	58.5	+4.1	58.00	+3.42
	Qwen3-VL-2B-Instruct	57.0	+2.7	49.2	-3.6	58.8	+2.0	**63.1**	+8.7	57.03	+2.45
	DocOpt	60.0	+5.7	**53.6**	+0.8	**64.6**	+7.8	59.4	+5.0	**59.40**	+4.83
OPEN-WEIGHTS											
Qwen3-Emb-0.6B	Direct Retrieval	46.3	0.0	39.8	0.0	38.6	0.0	33.4	0.0	39.53	0.0
	GPT-5.4-mini	54.5	+8.2	42.6	+2.8	43.5	+4.9	47.0	+13.6	46.92	+7.39
	Claude Haiku 4.5	43.4	-2.9	41.0	+1.2	30.0	-8.6	35.0	+1.6	37.34	-2.19
	Qwen3-VL-2B-Instruct	54.4	+8.1	45.8	+6.0	54.1	+15.5	55.8	+22.4	52.53	+13.00
	DocOpt	**58.5**	+12.2	**53.9**	+14.1	58.2	+19.6	58.8	+25.4	**57.35**	+17.83
	Retriever Fine-tuning	53.0	+6.7	47.8	+8.0	**65.0**	+26.4	59.3	+25.9	56.28	+16.75
	DocOpt - Joint Adaptation	55.2	+8.9	49.5	+9.7	52.5	+13.9	**60.1**	+26.7	54.33	+14.80
Qwen3-Emb-4B	Direct Retrieval	51.2	0.0	45.3	0.0	42.7	0.0	38.8	0.0	44.50	0.0
	GPT-5.4-mini	50.9	-0.3	46.7	+1.4	41.1	-1.6	52.7	+13.9	47.83	+3.33
	Claude Haiku 4.5	46.2	-5.0	45.4	+0.1	37.9	-4.8	44.1	+5.3	43.39	-1.11
	Qwen3-VL-2B-Instruct	58.8	+7.6	51.5	+6.2	57.1	+14.4	57.6	+18.8	56.25	+11.75
	DocOpt	58.9	+7.7	53.9	+8.6	60.0	+17.3	58.0	+19.2	57.70	+13.20
	Retriever Fine-tuning	58.9	+7.7	**55.0**	+9.7	**60.8**	+18.1	**62.2**	+23.4	**59.23**	+14.73
	DocOpt - Joint Adaptation	**59.6**	+8.4	51.5	+6.2	58.6	+15.9	**62.2**	+23.4	57.98	+13.48
Jina-Colbert-V2	Direct Retrieval	55.9	0.0	51.6	0.0	51.5	0.0	56.0	0.0	53.75	0.0
	GPT-5.4-mini	59.5	+3.5	48.9	-2.7	50.4	-1.1	52.5	-3.5	52.81	-0.94
	Claude Haiku 4.5	59.9	+4.0	49.1	-2.5	46.9	-4.6	56.3	+0.3	53.05	-0.71
	Qwen3-VL-2B-Instruct	57.8	+1.9	52.4	+0.8	56.7	+5.2	56.4	+0.4	55.83	+2.07
	DocOpt	58.9	+3.0	54.2	+2.6	63.1	+11.6	55.9	-0.1	58.03	+4.27
	Retriever Fine-tuning	59.3	+3.4	53.2	+1.6	59.4	+7.9	55.1	-0.9	56.75	+3.00
	DocOpt - Joint Adaptation	**61.2**	+5.3	**61.0**	+9.4	**72.6**	+21.1	**58.4**	+2.4	**63.30**	+9.55

Table 14: Visual document retrieval results. Direct Retrieval uses OCR. Deltas are absolute changes relative to Direct Retrieval.