GENERATE, THEN MATCH: LIGHTWEIGHT RELEVANCE MODELING FOR GENERATIVE RETRIEVAL

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative retrievers can generate relevant document identifiers yet rank the corresponding documents poorly, exposing a mismatch between generation likelihood and relevance. We introduce QIMATCH, a lightweight adaptation method that learns a relevance matching function over a retriever's existing query and identifier representations. With the generator and identifier index fixed, two shared projections learn to match queries with complete identifiers through contrastive training followed by document ranking refinement, updating only 1.18M parameters without an additional neural relevance teacher. At inference, QIMATCH combines matching scores with generation likelihood to rerank generated identifiers without encoding document text. On Natural Questions and MS MARCO, QIMATCH achieves the highest Recall@1 and MRR@10 among the compared methods for both product-quantized and title–URL identifiers. Across these four settings, it improves MRR@10 over supervised fine-tuning baselines by 3.08–12.33 percentage points. Analyses show that the learned matching function transfers to identifiers excluded from its supervision in both families, while fusion can correct ranking errors that neither score resolves alone. These findings demonstrate how learning to match existing representations enables effective relevance adaptation with a fixed generator.

1 INTRODUCTION

Generative retrieval returns documents by generating their identifiers directly from a query (Tay et al., 2022; Wang et al., 2022; Zhou et al., 2024b). Semantic identifiers represent documents as compact code sequences, enabling autoregressive search through a constrained identifier space (Sun et al., 2023; Zeng et al., 2024a). Generation likelihood governs both which identifiers are retrieved and how their documents are ranked. These decisions need not succeed together: a relevant document can enter the candidate list yet receive a lower likelihood than competing documents.

Supervised fine-tuning (SFT) already connects queries, documents, and identifiers. Indexing tasks associate document content with identifiers, while retrieval tasks associate real or synthetic queries with relevant identifiers (Zhuang et al., 2023; Zhou et al., 2024b; Mekonnen et al., 2025). These tasks learn query-conditioned representations through token prediction; final ranking then uses the accumulated likelihood of the identifier sequence. Figure 1 shows this mismatch: for a query about the lyricists of *The Phantom of the Opera*, the generator ranks the novel first and the relevant musical twelfth. The relevant document is already among the candidates, but the generation score favors a different interpretation of the same title. The example motivates adapting how the available candidates are compared for relevance.

Relevance-oriented post-training has advanced generative retrieval through listwise learning, preference optimization, and reinforcement learning (Tang et al., 2024; Mekonnen et al., 2025; Zhao et al., 2026). Other approaches combine generation with semantic or discriminative scoring (Lu et al., 2025; Ji et al., 2026), and analyses relate generative relevance computation to dense matching (Wu et al., 2024). An accompanying question is how much of this improvement requires updating the generator itself. Existing query states and identifier representations may already contain useful relevance information, but token-prediction training does not directly optimize their similarity

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.

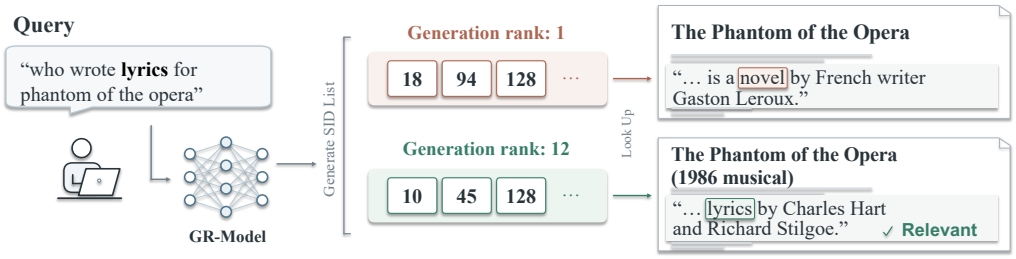


Figure 1: An example of ranking mismatch in generative retrieval (NQ). The query asks for lyricists, yet the novel ranks first and the relevant musical twelfth.

as a score of document relevance. This suggests adapting the comparison between them: a shared function can learn which query–identifier relationships distinguish relevant candidates from their competitors. Such adaptation reuses existing representations while preserving the generator and identifier assignments.

We introduce QIMATCH, which learns to compare the retriever's existing query and identifier representations for relevance. Adaptation trains two shared projections through candidate-level matching and document-ranking refinement, updating 1.18M parameters while preserving the generator and identifier index. The same architecture supports product-quantized (PQ) and title–URL (TU) identifiers, requiring no additional neural relevance teacher or online document-text encoding. On Natural Questions and MS MARCO, QIMATCH attains the highest Recall@1 and MRR@10 among compared methods in each identifier family. The learned matching score complements generation across both identifier families. Identifier-holdout experiments on PQ and TU show that the base matcher generalizes to identifiers excluded from its supervision, supporting a reusable relevance comparison over existing representations.

Our contributions are threefold:

- We identify relevance information underused by generation likelihood: learning to compare fixed query and identifier representations improves ranking within the same candidate set.
- We propose QIMATCH, which learns query–identifier relevance matching with 1.18M trainable parameters and achieves the highest Recall@1 and MRR@10 among the compared methods within both identifier families on both benchmarks.
- We show that matching complements generation and generalizes beyond supervised identifiers for both PQ and TU. Controlled interventions further identify how query information and identifier structure support the learned comparison.

2 RELATED WORK

Generative Retrieval and Semantic Identifiers. Generative retrieval maps queries to document identifiers within a sequence model (Tay et al., 2022; Wang et al., 2022; Zhou et al., 2024b). Query generation narrows the indexing–retrieval gap (Zhuang et al., 2023). Identifier designs include passage substrings (Bevilacqua et al., 2022), learned discrete codes (Sun et al., 2023), and complementary textual views (Li et al., 2023). RIPOR quantizes relevance-based document representations and applies prefix-oriented optimization (Zeng et al., 2024a). GLEN learns dynamic lexical identifiers with relevance supervision and ranks colliding documents using identifier logits (Lee et al., 2023). We retain existing identifier assignments and learn to compare their representations with queries.

Relevance Modeling in Generative Retrieval. Relevance modeling includes ranking objectives in LTRGR (Li et al., 2024), listwise training and calibration in ListGR (Tang et al., 2024), reward learning in GenRRL (Zhou et al., 2023), and direct preference optimization in DDRO (Mekonnen et al., 2025). Other work connects generative scoring to multi-vector dense retrieval (Wu et al., 2024). DOGR fine-tunes the retriever with query–document contrastive learning and fuses semantic scores with generation probabilities (Lu et al., 2025). GDR shares a query encoder across cluster generation and dense matching (Yuan et al., 2024), while UniGRF unifies retrieval and ranking for recommen-

dation (Zhang et al., 2025). UniGD combines fixed-codebook representations with adapter-based scoring; it freezes the shared decoder for adapter training before joint generative–discriminative optimization (Ji et al., 2026). QIMATCH keeps the generator fixed throughout adaptation, learning dual-projection matching over existing query and complete-identifier representations through candidate-level contrastive supervision and document-ranking refinement.

Search-Aware Scoring and Optimization. Sequence-model studies examine the interaction of training, normalization, and beam search (Wiseman & Rush, 2016; Goyal et al., 2019). For generative retrieval, theoretical analysis identifies limitations of corpus constraints and beam search (Wu et al., 2025). PAG uses look-ahead guidance (Zeng et al., 2024b), while PRO combines prefix retention optimization with geometric scoring (Chen et al., 2026). Further work develops token-level relevance feedback (Zhao et al., 2026) and examines the robustness of planning signals under query perturbations (Mekonnen et al., 2026). Our matching score is applied to completed identifiers, enabling direct comparisons with likelihood-based ranking on the same generated candidates.

3 PRELIMINARIES

Generative Retrieval. Given a query q and a document collection $\mathcal{D}$, the goal is to return documents ranked by relevance. A fixed mapping associates each document $d \in \mathcal{D}$ with an identifier $s(d) = (s_1, \ldots, s_L)$. We denote the set of valid identifiers by $\mathcal{S}$ and a length-t prefix by $s_{\leq t}$. An autoregressive retriever parameterized by θ assigns an identifier the log-likelihood

$$\ell_\theta(q, s) = \log p_\theta(s \mid q) = \sum_{t=1}^{L} \log p_\theta(s_t \mid q, s_{<t}). \tag{1}$$

We write $G_\theta(q, s)$ for the generator's ranking score, retaining its decoding-time length normalization where applicable. Our PQ implementation uses the accumulated score for fixed-length identifiers; TU uses the length-normalized score for variable-length identifiers.

Training for Memorization and Generalization. Retrieval training adapts a pretrained sequence-to-sequence model through corpus memorization and query generalization. For example, DDRO's SFT stage combines indexing and retrieval tasks (Mekonnen et al., 2025). Indexing pairs document passages or selected key terms with their identifiers to learn corpus–identifier associations (Zhou et al., 2024b). Retrieval pairs real queries and document-derived pseudo-queries with relevant identifiers, providing varied query expressions for learning query-to-identifier mappings (Zhuang et al., 2023). These tasks share a teacher-forced token cross-entropy objective that increases the likelihood of the target identifier. We denote the resulting SFT model by θ_0; it supplies the generator and representations for our subsequent matching adaptation.

Constrained Decoding. A prefix trie specifies the valid next tokens after each prefix. Beam search retains at most B prefixes at each step and returns a set of completed identifiers $\mathcal{C}_q \subseteq \mathcal{S}$. Retrieved identifiers are mapped back to documents; an identifier may correspond to multiple documents when the mapping has collisions. We distinguish the candidate set $\mathcal{C}_q$ from its ranking scores.

4 METHOD

QIMATCH augments an SFT retriever with a learned query–identifier matching function. We freeze the SFT generator at θ_0, retain the identifier assignments and validity trie, and learn a relevance score over complete identifiers. The generator provides query-conditioned states, which we compare with fixed representations of candidate identifiers. Figure 2 illustrates query–identifier matching and its use in ranking for both PQ and TU identifiers. Across both datasets and identifier families, adaptation trains only two shared projections.

Whole-Identifier Matching. Let $v(s)$ be a fixed representation of a complete identifier, prepared from the existing retriever or identifier index. This separates how an identifier is represented from how its relevance to a query is learned. For PQ, an identifier serves as a key to the code vectors associated with a document. Let $c_{t,k}$ denote the fixed code vector associated with token k at position t. For a PQ identifier $s = (s_1, \ldots, s_L)$, we concatenate the selected subvectors:

$$v(s) = [c_{1,s_1}; \ldots; c_{L,s_L}]. \tag{2}$$

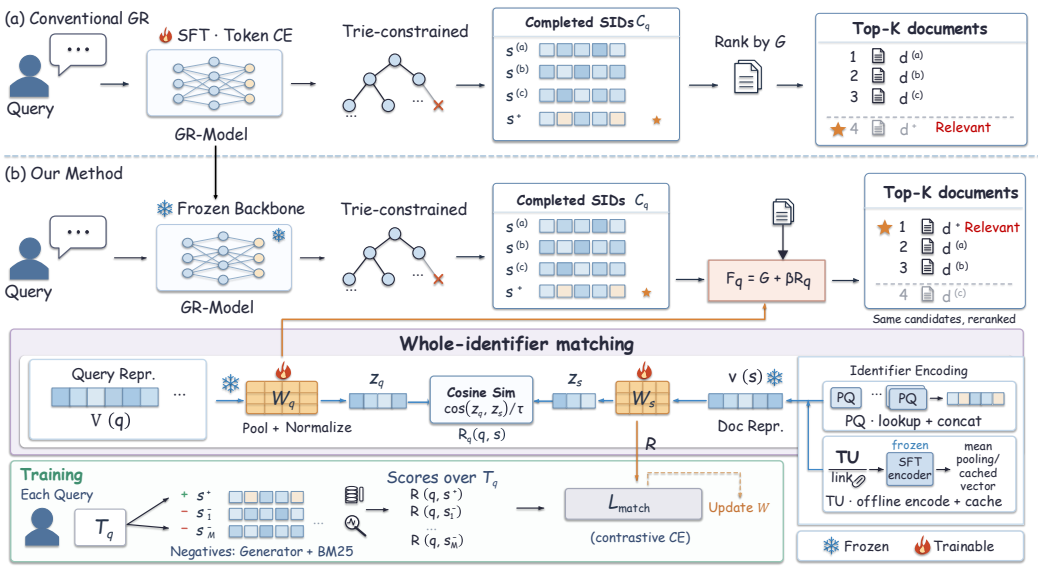


Figure 2: (a) Conventional GR uses token-level training and likelihood ranking. (b) QIMATCH freezes the SFT backbone, trains the matching projections with contrastive supervision, and reranks the G-generated candidates by $G + \beta R$.

For title–URL (TU) identifiers, $v(s)$ is the mean of the frozen SFT encoder states for the identifier's non-padding tokens, computed offline and cached. Both constructions represent the complete identifier without changing its document assignment.

The query-side representation is read from the frozen encoder. Let $H_{\theta_0,i}(q)$ denote its state at query position i, and let $\mathcal{I}_q$ contain the non-padding positions. We normalize their mean:

$$h(q) = \mathrm{norm}\left(\frac{1}{|\mathcal{I}_q|}\sum_{i\in\mathcal{I}_q} H_{\theta_0,i}(q)\right), \tag{3}$$

where $\mathrm{norm}(x) = x/\|x\|_2$. This query representation is shared by every candidate for the query.

The query states and identifier vectors need not be directly comparable in their original spaces. We learn bias-free projections on both sides and compare their directions in a shared space:

$$R_\phi(q, s) = \frac{\cos(W_q h(q), W_s v(s))}{\tau}, \tag{4}$$

where $\phi = \{W_q, W_s\}$ and $\tau > 0$ controls the scale of the matching logits. Learning both projections lets the matching function adapt the geometry on each side while preserving the backbone states and identifier vectors. The same pair of matrices processes every query and complete identifier, so the trainable parameter count does not grow with the collection size. This sharing also allows the learned comparison to be applied to identifiers absent from head supervision.

Learning from Retrieval Competition. We train the matching function on a relevant identifier and competing identifiers for the same query. The initial SFT generator supplies negatives that it retrieves for that query, while BM25 contributes candidates obtained through lexical retrieval. We combine the two sources, map documents to their identifiers, and remove known positive identifiers from the negative pool. The relevant identifier is included as the positive training target, whether or not the initial generator retrieves it. The negative pools are prepared from the fixed retriever and corpus and reused within each training stage.

For a training set $\mathcal{T}_q = \{s^+\} \cup \mathcal{N}_q$, the matching loss is

$$\ell_{\mathrm{match}}(q) = -\log \frac{\exp R_\phi(q, s^+)}{\sum_{s\in\mathcal{T}_q} \exp R_\phi(q, s)}. \tag{5}$$

Averaging over queries gives $\mathcal{L}_{\text{match}}$. The softmax compares the positive identifier against sampled competitors, following the contrastive training principle used in dense retrieval (Karpukhin et al., 2020). It uses the available relevance labels without an additional model to assign relevance scores. We initialize both projections to the identity and learn the matching geometry with this objective. We then refine the same projections with a document-ranking auxiliary loss:

$$\mathcal{L}_{\text{adapt}} = \mathcal{L}_{\text{match}} + \lambda \mathcal{L}_{\text{rank}}. \tag{6}$$

The auxiliary term accounts for the document positions occupied by competing identifiers. An identifier that expands to several documents can push the relevant document down by several positions, even though it contributes only one competitor to the matching softmax. Let $\mathcal{D}(s)$ be the ordered document set associated with identifier s, and let o_q be the zero-based position of the first relevant document within the positive identifier's list. We approximate its rank within the sampled competition set by

$$\widetilde{r}_q = 1 + o_q + \sum_{s^- \in \mathcal{N}_q} \min\big(|\mathcal{D}(s^-)|, 10\big)\, \sigma\left(\frac{R_\phi(q, s^-) - R_\phi(q, s^+)}{\tau_r}\right), \tag{7}$$

where σ is the sigmoid and $\tau_r = 1$. Each negative's contribution increases when it outscores the positive and is weighted by the number of document positions it can occupy, capped at the evaluation cutoff. The auxiliary loss $\mathcal{L}_{\text{rank}}$ averages $\log \widetilde{r}_q$ over examples with $o_q < 10$. Other examples retain the matching loss because reordering identifiers cannot bring those relevant documents into the top ten under the fixed within-identifier order. Only ϕ is updated in both stages, preserving the generator's representations, token probabilities, and identifier index. Appendix A.1 gives implementation details.

Combining Matching and Generation. At inference, the generator's score G_{θ_0} alone drives trie-constrained beam search to produce $\mathcal{C}_q$. We retain each completed identifier's generation score and evaluate its query–identifier compatibility, then rank by

$$F_\phi(q, s) = G_{\theta_0}(q, s) + \beta R_\phi(q, s), \qquad s \in \mathcal{C}_q. \tag{8}$$

The fixed weight $\beta \geq 0$ controls the influence of matching relative to likelihood. A candidate can move ahead of a higher-likelihood competitor when its matching advantage offsets the generation-score gap. Matching can reorder candidates while retaining likelihood evidence. The ordered identifiers are expanded into documents using the existing mapping and expansion policy.

5 Experiments

5.1 Experimental Setup

Datasets. We evaluate on MS MARCO document ranking (Bajaj et al., 2016) and Natural Questions (NQ320K) (Kwiatkowski et al., 2019), using the processed collections released with DDRO (Mekonnen et al., 2025). MS MARCO pairs web-search queries with documents; NQ pairs questions with Wikipedia pages. We evaluate all 808 MS MARCO and 7,830 NQ development queries.

Baselines. We compare term-based, dense, and generative retrieval in Table 1. Term-based baselines are BM25 (Robertson & Zaragoza, 2009) and DocT5Query (Nogueira & Lin, 2019); dense baselines are DPR (Karpukhin et al., 2020), ANCE (Xiong et al., 2021), RepBERT (Zhan et al., 2020), and Sentence-T5 (Ni et al., 2022). Generative baselines include DSI (Tay et al., 2022), DSI-QG (Zhuang et al., 2023), NCI (Wang et al., 2022), SEAL (Bevilacqua et al., 2022), Ultron (Zhou et al., 2024b), and MINDER (Li et al., 2023). Relevance-oriented baselines include LTRGR (Li et al., 2024), ROGER-NCI and ROGER-Ultron (Zhou et al., 2024a), GenRRL (Zhou et al., 2023), DDRO (Mekonnen et al., 2025), and TCA-GRPO (Zhao et al., 2026). DDRO's SFT retriever (Mekonnen et al., 2025) serves as the base model; Appendix A distinguishes published reference results from our checkpoint evaluations. Scoring-head controls additionally include UniGD-style (Ji et al., 2026) and DOGR-style (Lu et al., 2025) variants under the common protocol in Appendix B.3.

Models. We use the official T5-base SFT checkpoints (Raffel et al., 2020) for product-quantized (PQ) and title–URL (TU) identifiers, retaining each checkpoint's tokenizer and identifier index. In all four settings, we freeze the generator and train the same two-projection architecture through matching initialization and the refinement objective in Equation 6.

Table 1: Retrieval results on NQ320K and MS MARCO (%). Bold marks the best value within each PQ or TU group. SI: semantic IDs; NG: n-grams; Sum: summaries.

	NQ320K				MS MARCO			
Method (ID)	R@1	R@5	R@10	MRR@10	R@1	R@5	R@10	MRR@10
Term-based and dense retrieval								
BM25	14.06	36.91	47.93	23.60	18.94	42.82	55.07	29.24
DocT5Query	19.07	43.88	55.83	29.55	23.27	49.38	63.61	34.81
DPR	22.78	53.44	68.58	35.92	29.08	62.75	73.13	43.41
ANCE	24.54	54.21	69.08	36.88	29.65	63.43	74.28	44.09
RepBERT	22.57	52.20	65.65	35.13	25.25	58.41	69.18	38.48
Sentence-T5	22.51	52.00	65.12	34.95	27.27	58.91	72.15	40.69
Generative retrieval with other identifiers								
DSI (SI)	27.42	47.26	56.58	34.31	25.74	43.58	53.84	33.92
DSI-QG (SI)	30.17	53.20	66.37	38.85	28.82	50.74	62.26	38.45
NCI (SI)	32.69	55.82	69.20	42.84	29.54	57.99	67.28	40.46
SEAL (NG)	29.30	54.12	68.53	40.34	27.58	52.47	61.01	37.68
ROGER-NCI (SI)	33.20	56.34	69.80	43.45	30.61	59.02	68.78	42.02
MINDER (SI)	31.00	55.50	65.79	43.50	29.98	58.37	71.92	42.51
LTRGR (SI)	32.80	56.20	68.74	44.80	32.69	64.37	72.43	47.85
GenRRL (Sum)	36.32	57.42	71.49	46.31	33.23	64.48	75.80	46.62
Product-quantized identifiers (PQ)								
Ultron (PQ)	25.64	53.09	65.75	37.12	31.55	63.98	73.14	45.35
SFT (PQ)	40.24	58.88	64.48	48.25	30.19	62.50	71.16	43.57
DDRO (PQ)	48.10	62.64	66.84	54.32	31.19	63.00	71.91	44.41
TCA-GRPO (PQ)	50.02	64.09	67.36	56.10	32.42	61.88	71.16	44.85
QIMATCH (PQ)	**53.97**	**69.25**	**72.06**	**60.58**	**34.41**	**63.99**	**73.64**	**46.65**
Title–URL identifiers (TU)								
Ultron (TU)	33.78	54.20	67.05	42.51	29.82	60.39	68.31	42.53
ROGER-Ultron (TU)	35.90	55.59	69.86	44.92	33.07	63.93	75.13	46.35
GenRRL (TU)	35.79	56.49	**70.96**	45.73	33.01	63.62	74.91	45.93
SFT (TU)	38.62	54.15	57.93	45.24	36.63	59.78	68.19	46.53
DDRO (TU)	39.70	51.96	54.17	44.72	38.24	67.33	74.26	50.33
TCA-GRPO (TU)	40.63	52.42	55.39	45.53	39.60	68.19	75.37	51.10
QIMATCH (TU)	**50.26**	**65.35**	68.86	**56.83**	**43.07**	**71.16**	**79.46**	**54.56**

Evaluation. Following prior work, we evaluate on the benchmark development sets and report document-level Recall@1/5/10 and MRR@10. We compare PQ and TU methods within their respective identifier groups. Appendix A gives data and implementation details. Appendix B.1 presents additional controlled comparisons using disjoint model-selection and evaluation queries.

5.2 Main Results

Retrieval Performance. QIMATCH achieves the strongest results among the listed PQ-based methods across all four metrics on both NQ and MS MARCO (Table 1). Its advantage extends from the first returned document to the top-10 list, combining strong first-position accuracy with recall at greater ranking depths. Relative to the SFT rows in the table, PQ MRR@10 improves by 12.33 and 3.08 percentage points on NQ and MS MARCO. The TU variant also leads its group in R@1, R@5, and MRR@10 on both datasets, with corresponding MRR@10 gains of 11.59 and 8.03 points. The joint gains in Recall@1 and MRR@10 reflect stronger first-result relevance and ranking quality near the top of the list across both semantic-code and textual identifiers. GenRRL additionally uses DPR and LLaMA-13b relevance signals to train its reward model (Zhou et al., 2023); our adaptation uses the existing relevance labels and no additional neural relevance teacher.

Scoring-Head Design. To assess the dual-projection design, we compare it with query-only projection, a nearly parameter-matched UniGD-style MLP, and a DOGR-style query–document matcher (Table 2). We implement these scoring variants under our frozen-backbone protocol. All heads

Table 2: Scoring-head comparison under a common fixed training recipe. Values are fused MRR@10 (%); parameter counts refer to the trainable head.

		NQ		MS MARCO	
Scoring head	Params (M)	PQ	TU	PQ	TU
Dual projections (CE)	1.180	**59.67**	**53.92**	**43.85**	**53.08**
Query projection (CE)	0.590	57.99	52.70	41.80	52.73
UniGD-style MLP (CE)	1.183	57.88	51.99	42.21	51.94
DOGR-style q–doc (CE)	1.180	57.67	51.97	40.85	50.77

use a common cross-entropy objective, sixteen negatives, a fixed 20-epoch budget, and the same score-calibration rule (Appendix B.3). Dual-projection SID matching outperforms both query-only projection and the nearly parameter-matched MLP across all four settings. These comparisons support adapting both representation spaces with a simple shared matching function.

Training Stability and Refinement. Repeating all head-training stages with five seeds at fixed configurations gives MRR@10 standard deviations below one percentage point in all four settings (Appendix B.7). Removing only $\mathcal{L}_{\mathrm{rank}}$ under matched refinement budgets reduces MRR@10 by 0.06–0.47 percentage points, supporting a further improvement beyond the base matching objective (Appendix B.8).

Adaptation Cost. QIMATCH requires no generator fine-tuning: it updates only a 1.18M-parameter matching module while keeping the pretrained retriever frozen. Compared with full-generator updates in DDRO and TCA-GRPO, this yields substantially fewer trainable parameters and shorter training runs across both datasets and identifier types (Table 3). At inference, matching and ranking add a median of just 0.71–0.75 ms per query across the four settings. Appendix A.2 details the cost analysis and timing protocol.

Table 3: Trainable parameters and training runtime.

	Trainable (M)	NQ (min)	MS MARCO (min)
Method	PQ / TU	PQ / TU	PQ / TU
DDRO	257.02 / 222.88	113.5 / 191.0	117.1 / 120.1
TCA-GRPO	257.02 / 222.88	250.0 / 773.8	290.4 / 901.1
QIMATCH	**1.18 / 1.18**	**10.1 / 11.3**	**8.7 / 12.3**

6 FINDINGS

The main results show that a small matching module improves ranking with a fixed generator. To explain these gains and assess the transfer of the learned comparison, we address the following research questions:

1. **RQ1:** What makes relevance accessible in fixed representations?
2. **RQ2:** Why do generation and matching rank better together?
3. **RQ3:** Can matching transfer to identifiers excluded from its supervision?

6.1 READING OUT RELEVANCE FROM FIXED REPRESENTATIONS

We first ask how a small head extracts query-specific relevance from fixed representations. A uniform similarity increase would leave candidate rankings unchanged. We compare pretrained T5-base and retrieval-trained SFT encoders with identity or CE-trained projections, keeping PQ vectors, candidates, supervision, and budgets for the learned heads fixed. We also shuffle evaluation-query representations while preserving the trained heads and candidate vectors.

Figure 3(A) shows that trained projections make even generic pretrained states useful for matching, with SFT states supporting stronger rankings under the same budget.

Adaptation enlarges relevance margins. For each query, we fix its highest-G known positive SID, its highest-G negative SID, and a random negative from the same candidate set, then compare similarities before and after training. Figure 3(B) shows a shared increase in mean similarity, with larger gains for positives. The mean positive–highest-G negative margin increases by 0.147 on NQ

(A) Matching controls

Encoder / head	NQ	MS MARCO
Pretrained / identity	3.48	4.27
Pretrained / trained	48.53	26.92
SFT / identity	3.23	3.55
SFT / trained	**56.67**	**37.73**
SFT / shuffled query	3.11	1.92

Matching-only MRR@10 (%)

(B) Query–SID similarity

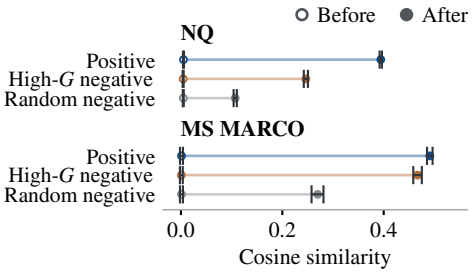


Figure 3: Learning relevance matching from fixed representations. **(A)** Matching-only MRR@10 of PQ representation controls. **(B)** Mean query–SID cosine similarities before and after training for fixed positive, highest-G negative, and random negative pairs. Black whiskers in (B) denote 95% query-bootstrap confidence intervals.

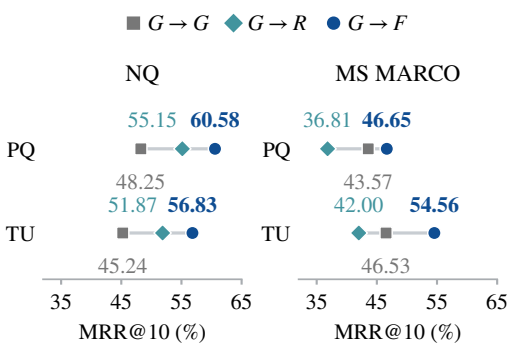
and 0.024 on MS MARCO. The learned comparison thus gives relevant identifiers a larger advantage over competing candidates.

The comparison is query-dependent. Figure 3(A) shows that query shuffling nearly removes the matching benefit. Its dependence on the query–candidate correspondence supports a learned relevance comparison beyond identifier preferences alone. Table 2 compares scoring-head designs; supplementary scoring and geometry results are provided in Appendix B.

6.2 Relevance Matching as a Ranking Correction

We next examine how generation likelihood complements learned relevance. Using the final matching heads, we compare $G \to G$, $G \to R$, and $G \to F$ on the same G-generated candidates, where $F = G + \beta R$. Figure 4(A) shows that fusion achieves the highest MRR@10 across both datasets and identifier families. On MS MARCO, matching alone trails generation, yet adding its score improves ranking for both identifier families.

Why retain the generation score? Figure 4(B) groups queries by whether each score places a relevant document first. On MS MARCO–TU, fusion retains 81.9% of the 149 correct G results missed by R, while recovering 66.7% of the 78 queries where R succeeds and G fails. The disagreement groups reveal complementary strengths: likelihood protects correct generation results, while matching supplies corrections that generation misses.



(A) Ranking performance



(B) Fusion decisions

Figure 4: Matching complements generation in aggregate and per query. **(A)** MRR@10 on shared G-generated candidates. **(B)** Fusion's top-1 hit rate grouped by the G/R top-1 outcomes (n: group size). Outlined cells highlight queries missed at rank one by both scores.

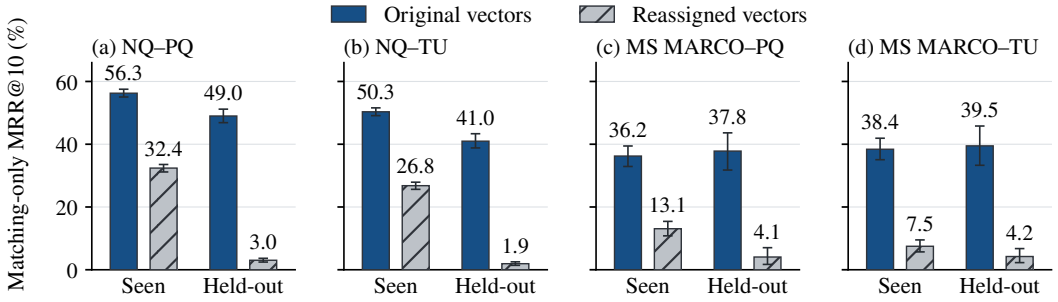


Figure 6: Original identifier representations support matching beyond supervised identifiers across PQ and TU. Bars show matching-only MRR@10 on seen and held-out identifiers; error bars denote 95% query-bootstrap intervals.

Recovering queries missed by both scores. The outlined cells in Figure 4(B) show that fusion also succeeds when both individual rankings miss at rank one. Fusion corrects 1,273 NQ–PQ and 79 MS MARCO–TU rank-one errors of G, of which 16.2% and 34.2% are also missed by R. Figure 5 makes this complementarity concrete: G favors indoor air pollution, while R favors compressed air. The relevant document ranks third under each score, but their combination suppresses both competitors and places it first. Score margins and rank-group results are in Appendix B.5; Appendix B.6 analyzes fusion weights.

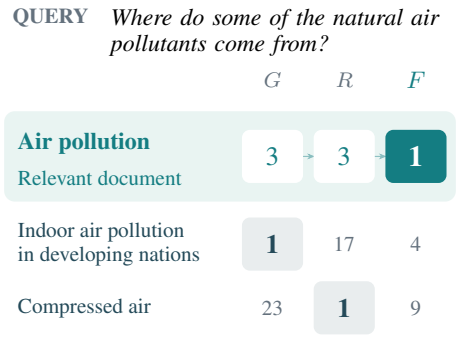


Figure 5: Fusion succeeds when both scores miss at rank one.

6.3 Learning a Transferable Matching Relation

We finally ask whether matching transfers to identifiers absent from head supervision. This tests whether adaptation can reuse identifier structure beyond the assignments observed during training. We withhold approximately 20% of complete PQ and TU identifiers from positive and negative supervision of the base matching stage. For PQ, every constituent codeword still occurs in the retained supervision; the held-out unit is the complete identifier. The generator and corpus index retain their original training exposure. Separate heads use either the original vectors or a fixed random reassignment of complete vectors among identifiers, with the same retained examples, negative pools, and training budget. Reassignment preserves the corpus vector set, while generation scores and document mappings remain unchanged.

Figure 6 shows higher held-out MRR@10 with original vectors than with random reassignment across both identifier families. The larger penalty on held-out than on seen identifiers supports a shared matching relation that reuses identifier structure beyond supervised assignments. Full matching and fusion results are provided in Appendix B.2.

7 Conclusion

We presented QIMatch, a lightweight adaptation method for addressing the mismatch between generation likelihood and relevance ranking. It learns two shared projections over fixed query and identifier representations and combines the resulting matching score with generation likelihood. On NQ and MS MARCO, QIMatch improves Recall@1 and MRR@10 for both PQ and TU identifiers using only 1.18M trainable parameters and no additional neural relevance teacher. Across all four settings, matching complements generation even when it is the weaker standalone ranker. Controlled studies show that the learned matching function is query-dependent and transfers to identifiers excluded from matching supervision, supporting reusable relevance structure in existing representations. These results demonstrate that learning to match existing representations can improve relevance ranking while preserving the generator.

AI USE STATEMENT

LLM-based assistants were used to help analyze experimental data, prepare figures and tables, and polish the main text. The authors take responsibility for the claims, code, results, references, and other content of this work, including materials produced with AI assistance.

ETHICS STATEMENT

We evaluate QIMATCH on the existing Natural Questions and MS MARCO benchmarks. The study does not involve recruiting human participants or collecting new personal data. As with other retrieval systems, rankings may reflect biases and uneven coverage in the underlying corpora and relevance annotations. These factors should be considered in downstream deployment.

REPRODUCIBILITY STATEMENT

The matching function and learning objectives are specified in Section 4. Section 5.1 and Appendix A describe the datasets, identifier representations, implementation settings, and evaluation protocol. The controlled-study protocols and training-seed results are provided in Appendices B.2 and B.7, respectively.

REFERENCES

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

Michele Bevilacqua, Giuseppe Ottaviano, Patrick Lewis, Scott Yih, Sebastian Riedel, and Fabio Petroni. Autoregressive search engines: Generating substrings as document identifiers. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 31668–31683. Curran Associates, Inc., 2022. doi: 10.52202/068431-2296. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/cd88d62a2063fdaf7ce6f9068fb15dcd-Paper-Conference.pdf.

Yufei Chen, Zihan Wang, Yubao Tang, Yukun Zhao, Maarten de Rijke, and Zhaochun Ren. Closing the indexing-decoding gap in multimodal generative retrieval via prefix retention optimization, 2026. URL https://arxiv.org/abs/2606.09241.

Kartik Goyal, Chris Dyer, and Taylor Berg-Kirkpatrick. An empirical investigation of global and local normalization for recurrent neural sequence models using a continuous relaxation to beam search. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 1724–1733, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1171. URL https://aclanthology.org/N19-1171/.

Shujie Ji, Yawei Kong, Yilin Zhao, Li Wang, Xialong Liu, and Peng Jiang. Unigd: A unified generative-discriminative framework for industrial retrieval, 2026. URL https://arxiv.org/abs/2608.03150.

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/.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019. doi: 10.1162/tacl_a_00276. URL https://aclanthology.org/Q19-1026/.

Sunkyung Lee, Minjin Choi, and Jongwuk Lee. GLEN: Generative retrieval via lexical index learning. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7693–7704, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.477. URL https://aclanthology.org/2023.emnlp-main.477/.

Yongqi Li, Nan Yang, Liang Wang, Furu Wei, and Wenjie Li. Multiview identifiers enhanced generative retrieval. 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. 6636–6648, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.366. URL https://aclanthology.org/2023.acl-long.366/.

Yongqi Li, Nan Yang, Liang Wang, Furu Wei, and Wenjie Li. Learning to rank in generative retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 8716–8723, 2024.

Penghao Lu, Xin Dong, Yuansheng Zhou, Lei Cheng, Chuan Yuan, and Linjian Mo. Dogr: Leveraging document-oriented contrastive learning in generative retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 24732–24740, 2025.

Kidist Amde Mekonnen, Yubao Tang, and Maarten de Rijke. Lightweight and direct document relevance optimization for generative information retrieval. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1327–1338, 2025.

Kidist Amde Mekonnen, Yongkang Li, Yubao Tang, Simon Lupart, and Maarten de Rijke. Lost in decoding? reproducing and stress-testing the look-ahead prior in generative retrieval. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2994–3005, 2026.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 1864–1874, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.146. URL https://aclanthology.org/2022.findings-acl.146/.

Rodrigo Nogueira and Jimmy Lin. From doc2query to docTTTTTquery. Technical report, 2019. URL https://cs.uwaterloo.ca/~jimmylin/publications/Nogueira_Lin_2019_docTTTTTquery-v2.pdf.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009. doi: 10.1561/1500000019. URL https://doi.org/10.1561/1500000019.

Weiwei Sun, Lingyong Yan, Zheng Chen, Shuaiqiang Wang, Haichao Zhu, Pengjie Ren, Zhumin Chen, Dawei Yin, Maarten de Rijke, and Zhaochun Ren. Learning to tokenize for generative retrieval. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=UKd6dpVGdu.

Yubao Tang, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Wei Chen, and Xueqi Cheng. Listwise generative retrieval models via a sequential learning process. *ACM Trans. Inf. Syst.*, 42(5), April 2024. ISSN 1046-8188. doi: 10.1145/3653712. URL https://doi.org/10.1145/3653712.

Yi Tay, Vinh Q. Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, Tal Schuster, William W. Cohen, and Donald Metzler. Transformer memory as a differentiable search index. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=Vu-B0clPfq.

Yujing Wang, Yingyan Hou, Haonan Wang, Ziming Miao, Shibin Wu, Qi Chen, Yuqing Xia, Chengmin Chi, Guoshuai Zhao, Zheng Liu, Xing Xie, Hao Sun, Weiwei Deng, Qi Zhang, and Mao Yang. A neural corpus indexer for document retrieval. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 25600–25614. Curran Associates, Inc., 2022. doi: 10.52202/068431-1856. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/a46156bd3579c3b268108ea6aca71d13-Paper-Conference.pdf.

Sam Wiseman and Alexander M. Rush. Sequence-to-sequence learning as beam-search optimization. In Jian Su, Kevin Duh, and Xavier Carreras (eds.), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1296–1306, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1137. URL https://aclanthology.org/D16-1137/.

Shiguang Wu, Wenda Wei, Mengqi Zhang, Zhumin Chen, Jun Ma, Zhaochun Ren, Maarten de Rijke, and Pengjie Ren. Generative retrieval as multi-vector dense retrieval. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pp. 1828–1838, 2024.

Shiguang Wu, Zhaochun Ren, Xin Xin, Jiyuan Yang, Mengqi Zhang, Zhumin Chen, Maarten de Rijke, and Pengjie Ren. Constrained auto-regressive decoding constrains generative retrieval. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2429–2440, 2025.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=zeFrfgyZln.

Peiwen Yuan, Xinglin Wang, Shaoxiong Feng, Boyuan Pan, Yiwei Li, Heda Wang, Xupeng Miao, and Kan Li. Generative dense retrieval: Memory can be a burden. In Yvette Graham and Matthew Purver (eds.), *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2835–2845, St. Julian's, Malta, March 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.eacl-long.173. URL https://aclanthology.org/2024.eacl-long.173/.

Hansi Zeng, Chen Luo, Bowen Jin, Sheikh Muhammad Sarwar, Tianxin Wei, and Hamed Zamani. Scalable and effective generative information retrieval. In *The Web Conference 2024*, 2024a. URL https://openreview.net/forum?id=7fw6EAxUI7.

Hansi Zeng, Chen Luo, and Hamed Zamani. Planning ahead in generative retrieval: Guiding autoregressive generation through simultaneous decoding. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 469–480, 2024b.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Min Zhang, and Shaoping Ma. Repbert: Contextualized text embeddings for first-stage retrieval, 2020. URL https://arxiv.org/abs/2006.15498.

Luankang Zhang, Kenan Song, Yi Quan Lee, Wei Guo, Hao Wang, Yawen Li, Huifeng Guo, Yong Liu, Defu Lian, and Enhong Chen. Killing two birds with one stone: Unifying retrieval and ranking with a single generative recommendation model. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2224–2234, 2025.

Xinpeng Zhao, Yang Liu, Ran Chen, Xinyu Ma, Daiting Shi, Pengjie Ren, Zhumin Chen, Zhaochun Ren, and Xin Xin. Token-level credit assignment optimization for generative document retrieval. *arXiv preprint arXiv:2608.12049*, 2026.

Yujia Zhou, Zhicheng Dou, and Ji-Rong Wen. Enhancing generative retrieval with reinforcement learning from relevance feedback. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12481–12490, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.768. URL https://aclanthology.org/2023.emnlp-main.768/.

Yujia Zhou, Jing Yao, Zhicheng Dou, Yiteng Tu, Ledell Wu, Tat-Seng Chua, and Ji-Rong Wen. ROGER: Ranking-oriented generative retrieval. *ACM Transactions on Information Systems*, 42(6):155:1–155:25, 2024a. doi: 10.1145/3603167. URL https://doi.org/10.1145/3603167.

Yujia Zhou, Jing Yao, Ledell Wu, Zhicheng Dou, and Ji-Rong Wen. Webultron: An ultimate retriever on webpages under the model-centric paradigm. *IEEE Transactions on Knowledge and Data Engineering*, 36(9):4996–5006, 2024b. doi: 10.1109/TKDE.2023.3332858.

Shengyao Zhuang, Houxing Ren, Linjun Shou, Jian Pei, Ming Gong, Guido Zuccon, and Daxin Jiang. Bridging the gap between indexing and retrieval for differentiable search index with query generation, 2023. URL https://arxiv.org/abs/2206.10128.

A Additional Experimental Details

Datasets and Identifiers. We use the processed NQ320K and MS MARCO collections released with DDRO (Mekonnen et al., 2025), retaining the document identifiers, tokenizers, and relevance annotations. Table 4 summarizes the collections and matching-training examples. Each PQ identifier has 24 codes, with 256 codewords per position and 32 dimensions per codeword, yielding a 768-dimensional vector by concatenation. For TU, we encode the released identifier token sequence with the original SFT encoder, mean-pool its non-padding states, and cache the resulting 768-dimensional vector. These identifier vectors remain fixed during head training.

Table 4: Collection sizes and examples used for matching adaptation. Training pairs count query–identifier examples; development counts refer to evaluation queries.

Dataset	Documents	Training pairs	Development queries
NQ	109,739	303,276	7,830
MS MARCO	319,927	362,916	808

Training Examples. Each example contains one relevant identifier and 16 distinct negatives, comprising eight generator-retrieved and eight BM25-retrieved identifiers. Known positive identifiers are excluded from the negative pool. BM25 candidates are sampled across rank strata within the top 1,000 results, using $k_1 = 0.82$ and $b = 0.68$ with Porter stemming. Generator candidates come from the corresponding PQ or TU SFT retriever. The competition sets and frozen features are prepared offline and reused within each training stage.

Implementation Details. Both projections are bias-free 768×768 matrices, initialized to the identity. We train the matching heads with cross-entropy initialization followed by document-ranking refinement, keeping the SFT backbone and identifier vectors fixed. For refinement, we use AdamW with a learning rate of 10^{-4} and a global batch size of 128. The matching temperature is $\tau = 0.05$, and dropout is disabled. The auxiliary weight λ is 0.1 for NQ–PQ and MS MARCO–TU, and 0.3 for NQ–TU and MS MARCO–PQ. The main experiments use eight NVIDIA H20 GPUs and seed 42; Appendix B.7 reports variation across five seeds.

Generation Scores and Document Ranking. For PQ, the accumulated generation score is recovered from the beam score using the common identifier length of 24. For TU, we retain the generation implementation's sequence score with length penalty 1, including its effective generated-length convention. The fusion weights therefore operate on different score scales across the two encodings. Search returns 80 completed identifiers; matching changes their order before document expansion. Within an identifier, document order follows the released mapping on NQ and sorted document identifiers on MS MARCO. Score ties retain the original candidate order, duplicate documents are removed, and the output is truncated to 100 documents. Recall@k is the fraction of annotated relevant documents in the first k positions, averaged over queries. MRR@10 averages the reciprocal rank of the first relevant document, assigning zero when none occurs in the top ten.

Baseline Sources. External baseline values in Table 1 are taken from the tables of TCA-GRPO (Zhao et al., 2026), including its DDRO reproduction and MS MARCO–PQ SFT reference. The NQ–PQ and TU SFT entries report our evaluation of the official base checkpoints, and the QIMatch rows use our frozen-generator matching heads. DDRO and TCA-GRPO state that their GenRRL and ROGER entries are sourced from the original papers because public implementations were unavailable (Mekonnen et al., 2025; Zhao et al., 2026). GenRRL's entry in the TU group corresponds to its published URL-identifier variant. All controlled $G/R/F$ comparisons use scores recomputed on the same candidates within each setting. Reported gains over the main-table SFT baselines use the values displayed in that table.

A.1 Document-Ranking Auxiliary Implementation

The soft document-rank proxy is defined in Equation 7. Its document counts and within-identifier offsets follow the fixed expansion rules above, and it uses the same sampled negatives as the matching loss. We average $\log \widetilde{r}_q$ over examples with $o_q < 10$ within each worker's minibatch; examples with larger offsets retain the matching CE loss. The auxiliary objective uses scalar operations on

matching logits and the existing identifier mapping, adding no trainable parameters. We scale the combined loss by 0.1 during refinement.

A.2 Adaptation Work Accounting

We count post-SFT optimization separately from candidate and feature preparation. For DDRO, n_D counts processed preferred/rejected pairs; for TCA-GRPO, n_T counts query-group exposures with m candidates; for QIMatch, n_M counts query exposures with K competing identifiers including the positive. These counts include repeated epochs. For our method, n_M includes both CE initialization and subsequent refinement exposures.

Training-Run Time. Table 3 reports wall-clock times on eight H20 GPUs for the complete training runs of the selected configurations, including loading, intermediate evaluation, and saving. For QIMatch, we sum initialization and refinement; offline candidate and feature preparation and other hyperparameter trials are excluded. These are complete-run times rather than times to the selected checkpoint.

Leading Matrix Products. A dense vocabulary projection with width d, vocabulary size V, and output length L requires $6LdV$ floating-point operations for its forward product and two backward products, counting a multiply–add as two operations. $C_\theta = 6LdV$ counts this component; transformer layers, query encoding, and candidate generation add further work. DDRO processes a preferred and a rejected sequence, while TCA-GRPO processes m generated sequences (Mekonnen et al., 2025; Zhao et al., 2026). For PQ, TCA-GRPO uses $m = 8$ and no reference-policy KL penalty.

The PQ matching implementation evaluates both projections for each of the K candidates. With fixed inputs, each projection requires a forward product and a parameter-gradient product, giving $C_\phi \simeq 8Kd^2$ for the leading dense products. Normalization, the matching loss, the $O(K)$ auxiliary ranking operations, and optimizer updates add lower-order work. For $d = 768$, $K = 17$, $L = 24$, and $V = 38{,}272$, these products cost approximately 0.080 GFLOPs per matching query, compared with 8.47 GFLOPs for the vocabulary projections of a DDRO pair and 33.86 GFLOPs for an eight-candidate TCA-GRPO group. The corresponding ratios are approximately $106\times$ and $422\times$ and quantify the stated matrix products per optimization unit. The corresponding leading-product totals are $2n_DC_\theta$, mn_TC_θ, and n_MC_ϕ; end-to-end adaptation cost also includes the other operations and preparation. Table 3 reports observed training-run times separately from these operation counts.

Preparation. Our preparation comprises generator-derived and BM25 negative pools and a frozen query-feature cache. PQ identifier vectors come from codeword lookup; TU additionally requires one offline encoder pass over each unique identifier and storage of its cached vector. DDRO constructs BM25-based preference pairs; TCA-GRPO caches gold trajectories and generates candidate groups during training. At inference, the matching head uses the query states and cached candidate vectors without an online document-text encoder.

Incremental Inference Time. We time 32 evenly spaced evaluation queries per setting serially on a single H20 GPU with models and SID vectors resident and no background GPU workload on the node. Each setting is measured three times, with two warm-up queries excluded from each run; we report the median across queries of their per-query median times over the three runs. Timing starts after query encoding and beam search and covers candidate-vector lookup, matching, score fusion, sorting, and document expansion. The median increments are 0.710/0.708 ms for NQ–PQ/TU and 0.711/0.750 ms for MS MARCO–PQ/TU. Tokenization, model loading, and offline vector preparation are outside this interval.

B SUPPLEMENTARY ANALYSES

B.1 CONTROLLED MODEL SELECTION

To test whether the gains persist under disjoint selection and evaluation queries, we rerun QIMATCH, DDRO, and TCA-GRPO from the same official SFT checkpoints with common training queries, identifier mappings, and beam width 80. Fixed text- and length-stratified halves of the development queries separate selection from evaluation: 3,915/3,915 for NQ and 404/404 for MS MARCO. All checkpoint and hyperparameter choices use selection-half MRR@10 and are fixed before evaluation. With 1.18M trainable parameters, QIMATCH achieves higher MRR@10 than DDRO and TCA-GRPO across all four controlled settings (Table 5). This split separates model selection from evaluation within the present comparison. The full development sets had been used during earlier method development, while the controlled comparison fixes all model and hyperparameter choices using only the selection half before evaluation.

Table 5: MRR@10 (%) on the evaluation half after model selection on the other half.

Setting	SFT	DDRO	TCA-GRPO	QIMATCH
NQ (PQ)	49.12	52.80	56.36	**61.28**
NQ (TU)	45.62	46.16	47.40	**56.72**
MS MARCO (PQ)	45.78	46.12	47.48	**48.35**
MS MARCO (TU)	47.47	50.40	50.50	**55.99**

B.2 CONTROLLED STUDIES OF THE BASE MATCHING STAGE

Representation and Query Controls. The pretrained-encoder control uses original T5-base weights (Raffel et al., 2020) with the same tokenized inputs and pooling rule as the SFT encoder. Both encoders remain fixed. Identity-initialized heads use the same examples, negative pools, optimizer, and fixed training budget within each dataset. These representation controls train the base matching objective $\mathcal{L}_{\text{match}}$, before document-ranking refinement. Identity controls leave the projections untrained. Query shuffling uses a fixed permutation of evaluation-query vectors while retaining the trained head and generated candidates.

Identifier Holdout. A deterministic split withholds approximately 20% of complete identifiers from head supervision in each identifier family. We group training examples by model-visible query tokens and exclude groups with a known positive in the held-out set. Held-out identifiers are also removed from negative pools, which are refilled from the remaining generator and BM25 candidates. Examples without sixteen distinct valid negatives are dropped identically across arms. The same exclusion rule is applied to PQ identifiers and TU token sequences, and the original-vector and reassignment arms share retained examples, negative pools, and training budgets. Every constituent PQ codeword remains present in retained supervision. For complete-vector reassignment, vectors assigned to held-out identifiers never enter head training. All arms use the same CE recipe within each dataset; the generator and cached identifier vectors remain fixed. The generator and corpus index retain their original corpus exposure; this control measures transfer to identifiers without direct matching-head supervision.

Diagnostic Evaluation Protocol. A fixed query hash assigns approximately one fifth of the development queries to score calibration and leaves the remainder for diagnostic evaluation. For RQ3, calibration uses queries whose relevant identifiers were seen during head training; held-out queries are excluded. Trained heads and fusion weights are fixed before evaluation on the seen and held-out partitions. The error bars in Figure 6 are 95% query-bootstrap intervals from 2,000 resamples with the trained heads fixed. Table 7 reports both seen and held-out results.

B.3 SCORING-HEAD COMPARISON PROTOCOL

We compare dual projections, query-only projection with $W_s = I$, and a UniGD-style frozen MLP scorer (Ji et al., 2026). At the time of our experiments, neither UniGD nor DOGR had released an open-source implementation. We adapt their published scoring principles as controls under the shared frozen-backbone protocol. These SID-based heads use the same frozen SFT features, sixteen

Table 6: Learning relevance matching from fixed representations (CE stage, PQ). Values are MRR@10 (%) on diagnostic queries with identical generated candidates. The final row alters the trained SFT head's query input.

	NQ		MS MARCO	
Representation and head	R only	F	R only	F
G-only baseline	48.43		44.86	
Pretrained T5 + identity head	3.48	48.44	4.27	44.77
Pretrained T5 + trained head	48.53	58.36	26.92	46.65
Retrieval SFT + identity head	3.23	48.39	3.55	44.99
Retrieval SFT + trained head	**56.67**	**59.70**	**37.73**	**47.01**
Shuffled queries	3.11	48.45	1.92	44.71

Table 7: Identifier-holdout results for the base matching stage. Values are MRR@10 (%) on seen and held-out diagnostic queries; R and F use the same generated candidates.

		NQ		MS MARCO	
Identifier representation	Partition	R only	F	R only	F
PQ identifiers					
G only	Seen	–	47.98	–	44.45
	Held out	–	49.59	–	46.43
Original vectors	Seen	56.26	58.88	36.23	46.59
	Held out	48.99	53.95	37.82	46.22
Reassigned vectors	Seen	32.40	54.55	13.08	45.81
	Held out	3.03	42.85	4.06	41.87
TU identifiers					
G only	Seen	–	45.36	–	47.43
	Held out	–	44.27	–	45.13
Original vectors	Seen	50.34	55.74	38.40	52.50
	Held out	40.97	50.33	39.48	51.45
Reassigned vectors	Seen	26.79	50.85	7.48	48.61
	Held out	1.93	37.77	4.22	42.35

negatives, and beam-80 candidates, with seed 42, AdamW at 10^{-4}, global batch 128, and a fixed 20-epoch budget evaluated at the final checkpoint. The MLP uses two $768 \to 512 \to 256$ adapters and a $512 \to 256 \to 1$ concatenation scorer, matching the dual projections' parameter count within 0.3%. We train it with the same $0.1\mathcal{L}_{\text{match}}$ objective. For this comparison, every head uses $\beta = \beta_0$, with $\beta_0 = s_G / \max(s_R, 10^{-6})$, where each scale is the median per-query score standard deviation on the same 1,024 unlabeled training queries.

We also evaluate a DOGR-style query–document matching control (Lu et al., 2025), replacing the candidate-side SID features with cached, mean-pooled non-padding document states from the frozen SFT encoder. Two bias-free 768×768 projections give the same 1.180M trainable parameters as the dual-projection matcher, using the CE objective, training budget, and score-calibration rule above.

Under the fixed CE protocol, dual-projection SID matching leads all four settings, including against the equally parameterized query–document control in Table 2.

B.4 Scoring the Same Candidates across PQ and TU

Table 8 expands Figure 4 with recall and query-level rank-one changes. For each main-method head, we retain its generated candidates, generation scores, document expansion order, and fusion weight, then compare generation, matching, and fusion. Recovered and lost counts are measured relative to generation-only ranking: a recovery moves an annotated relevant document to rank one, while a loss displaces a previously correct rank-one result. Fusion recovers rank-one misses while preserving more than 90% of G's correct top results in every setting.

Table 8: Generation, matching, and fusion on the same candidates in each setting. Metrics are percentages; recovered/lost counts refer to rank-one changes relative to G. The MS MARCO–PQ G R@1 of 30.20 is recomputed on these candidates; Table 1 reports the external SFT reference of 30.19 from TCA-GRPO.

Setting	Score	R@1	R@5	R@10	MRR@10	Recovered / lost
NQ (PQ)	G	40.24	58.88	64.48	48.25	–
	R	46.76	66.55	70.98	55.15	1267 / 757
	F	**53.97**	**69.25**	**72.06**	**60.58**	1273 / 198
NQ (TU)	G	38.62	54.15	57.93	45.24	–
	R	44.37	61.79	66.18	51.87	1236 / 786
	F	**50.26**	**65.35**	**68.86**	**56.83**	1174 / 263
MS MARCO (PQ)	G	30.20	62.50	71.16	43.57	–
	R	23.64	55.45	67.95	36.81	62 / 115
	F	**34.41**	**63.99**	**73.64**	**46.65**	56 / 22
MS MARCO (TU)	G	36.63	59.78	68.19	46.53	–
	R	27.85	61.39	73.02	42.00	78 / 149
	F	**43.07**	**71.16**	**79.46**	**54.56**	79 / 27

Table 9: Document ranking by the first relevant document's generation rank. Each score reports MRR@10 (M), Recall@1 (R1), and Recall@10 (R10), in percent. Pool membership is determined before top-100 truncation. Displayed groups contain a relevant document in the generated pool; absent queries are unchanged by reranking.

			G			R			F		
Setting	G rank	n	M	R1	R10	M	R1	R10	M	R1	R10
NQ–PQ	1	3151	100.0	100.0	100.0	84.4	76.0	98.7	96.5	93.7	99.9
	2–5	1459	38.8	0.0	100.0	65.0	50.0	92.8	72.7	55.2	98.6
	6–10	439	13.7	0.0	100.0	59.6	45.8	88.2	61.9	45.6	96.6
	>10, in pool	847	0.0	0.0	0.0	53.1	39.7	83.4	43.8	31.5	74.5
NQ–TU	1	3024	100.0	100.0	100.0	81.6	74.0	94.7	94.3	91.3	98.7
	2–5	1216	39.3	0.0	100.0	61.7	47.1	88.5	67.4	49.6	95.2
	6–10	296	13.5	0.0	100.0	46.0	34.5	74.0	48.1	34.1	87.5
	>10, in pool	1920	0.0	0.0	0.0	36.8	29.2	53.3	33.1	24.5	51.5
MS–PQ	1	244	100.0	100.0	100.0	67.1	52.9	95.1	95.0	91.0	100.0
	2–5	261	37.8	0.0	100.0	38.5	19.2	83.5	47.8	19.9	99.6
	6–10	70	13.5	0.0	100.0	17.1	5.7	62.9	20.1	4.3	90.0
	>10, in pool	117	0.0	0.0	0.0	18.0	6.8	47.0	5.5	0.9	23.9
MS–TU	1	296	100.0	100.0	100.0	65.1	49.7	91.6	94.2	90.9	99.7
	2–5	187	37.9	0.0	100.0	37.9	19.8	83.4	52.7	25.7	98.4
	6–10	68	13.4	0.0	100.0	36.7	22.1	72.1	36.8	17.6	86.8
	>10, in pool	182	0.0	0.0	0.0	27.9	14.3	62.6	21.2	10.4	57.1

B.5 Generation Rank and Fusion Repairs

We partition each full development set by the first relevant document's rank under G: 1, 2–5, 6–10, beyond 10 within the candidate pool, or absent from the pool. The frozen main heads, fusion weights, beam-80 candidates, identifier mappings, stable score sorting, stored collision-bucket order, and first-occurrence document deduplication match the main evaluation. Group assignment uses the complete expanded document pool before top-100 truncation; the reported metrics retain their original cutoffs. Each query here has one annotated relevant document.

A repair to top k moves a G miss at cutoff k into F's top k; its denominator includes all such G misses, including pool absence. A preserved G rank-one hit remains correct under F, with all G rank-one hits as the denominator. Of all top-1 repairs, 40.0% on NQ–TU start below the top ten under G, compared with 1.8% on MS MARCO–PQ, where 92.9% start at ranks 2–5. The gains combine local corrections and deeper promotions within the fixed candidate pool. For NQ–PQ, NQ–TU, MS MARCO–PQ, and MS MARCO–TU, respectively, 14, 1,288, 27, and 38 queries have a relevant document below G's top 100 but inside its full pool, whereas 1,934, 1,374, 116, and 75 are truly absent. Expansion can place documents deep in the ranking through intervening

Table 10: Fusion recoveries and preservation relative to G. Each entry gives the event count / eligible query count (percent). Recovery denominators include all G misses at the stated cutoff, including queries missing from the pool; preservation denominators contain G rank-one hits.

Setting	Recovered G errors at 1	Recovered G errors at 10	Preserved G top-1 hits
NQ–PQ	1273/4679 (27.2)	631/2781 (22.7)	2953/3151 (93.7)
NQ–TU	1174/4806 (24.4)	989/3294 (30.0)	2761/3024 (91.3)
MS–PQ	56/564 (9.9)	28/233 (12.0)	222/244 (91.0)
MS–TU	79/512 (15.4)	104/257 (40.5)	269/296 (90.9)

Table 11: Complementary score margins for the NQ–PQ natural-air-pollutants query. Scores use the main head and fixed $\beta = 66.833522$; document ranks use full expansion. Only *Air pollution* is annotated relevant.

Document	G	R	βR	F	Rank G	Rank R	Rank F
Air pollution	-51.196	4.589	306.695	255.498	3	3	1
Indoor air pollution in developing nations	-42.956	2.909	194.444	151.488	1	17	4
Compressed air	-336.623	5.864	391.882	55.258	23	1	9

collision buckets; these document ranks are not identifier beam positions. Fusion changes neither pool membership nor the order within a collision bucket.

For the natural-air-pollutants example, the annotated article explicitly covers volcanic emissions, dust, and wildfires. Against the G winner, its weighted matching advantage of 112.250 exceeds its generation deficit of 8.240; against the R winner, its generation advantage of 285.427 exceeds its weighted matching deficit of 85.187. The example illustrates complementary margins, while the full group statistics establish their aggregate effect.

B.6 The Effect of Matching Weight

For each setting, we select β^* on the selection half and freeze the checkpoint, candidates, and score arrays before evaluation. We then vary the matching weight by a multiplicative factor and recompute document ranking only on the disjoint evaluation half. Figure 7 shows that the selection-half β^* attains the highest evaluation-half MRR@10 among the plotted weights in all four PQ/TU settings. Larger departures from this operating point increasingly give up either matching corrections or useful generation preferences.

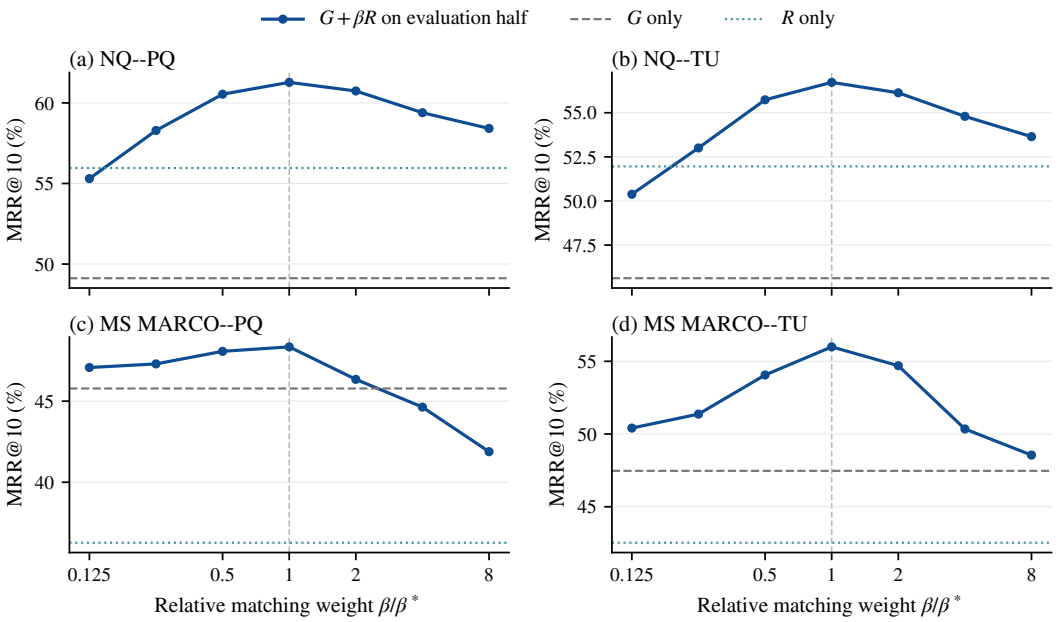


Figure 7: Evaluation-half sensitivity to the fusion weight selected on the disjoint selection half. β^* and the checkpoint are fixed using selection-half MRR@10; curves and horizontal references are recomputed on evaluation-half queries for all four PQ/TU settings. The vertical guide marks β^*.

B.7 Sensitivity to Training Seeds

We repeat all head-training stages from identity initialization with five seeds, fixing the main-table hyperparameters, stage budgets, negative pools, and fusion weights. The SFT generator, cached features, and beam-80 candidates remain fixed; each run is evaluated once on the full development set without further selection. Table 12 shows modest variation across all four settings, with MRR@10 standard deviations below one percentage point. These results quantify training-seed variability at fixed configurations.

Table 12: Retrieval performance over five training seeds: mean ± sample standard deviation (%).

Setting	R@1	R@5	R@10	MRR@10
NQ (PQ)	53.85 ± 0.14	69.21 ± 0.06	72.10 ± 0.08	60.57 ± 0.07
MS MARCO (PQ)	34.53 ± 0.12	64.06 ± 0.11	73.66 ± 0.10	46.68 ± 0.10
NQ (TU)	50.32 ± 0.13	65.39 ± 0.09	68.97 ± 0.20	56.93 ± 0.12
MS MARCO (TU)	43.02 ± 0.11	71.14 ± 0.14	79.43 ± 0.06	54.59 ± 0.09

B.8 Effect of Document-Ranking Refinement

To isolate the auxiliary objective defined in Section 4, we remove $\mathcal{L}_{\text{rank}}$ during the final refinement stage. Both variants start from identical CE-trained projections and use the same training examples, negative pools, optimizer, seed, and number of updates. We keep the generator, candidates, document expansion rules, and each setting's main-model fusion weight fixed. This comparison measures the auxiliary loss's contribution at the same training budget and inference configuration.

Table 13 shows higher Recall@1 and MRR@10 with the auxiliary objective in all four settings. The results support document-ranking refinement as an additional improvement on top of the relevance learned by the base matching loss.

Table 13: Removing the auxiliary loss under matched refinement budgets and fixed fusion weights. Values are percentages.

	Without $\mathcal{L}_{\text{rank}}$		QIMATCH	
Setting	R@1	MRR@10	R@1	MRR@10
NQ (PQ)	53.87	60.52	**53.97**	**60.58**
NQ (TU)	49.89	56.36	**50.26**	**56.83**
MS MARCO (PQ)	34.03	46.43	**34.41**	**46.65**
MS MARCO (TU)	42.82	54.31	**43.07**	**54.56**

B.9 GEOMETRY OF THE LEARNED MATCHING SPACE

We reuse the fixed SFT query states, PQ codebook vectors, and CE-trained heads from the representation controls. The geometry analysis uses the same diagnostic evaluation remainder, retaining queries with a positive SID and at least two distinct negative SIDs in the generated candidates: 4,735 NQ and 561 MS MARCO queries. For each query, we fix the highest-G positive SID, the highest-G SID outside its known positives, and one uniformly sampled remaining negative SID. The random draw is fixed by query ID and shared before and after adaptation. We compare normalized original vectors with normalized projected vectors, using cosine similarities before temperature scaling. Figure 3(B) reports mean similarities with 95% intervals from 5,000 query-bootstrap resamples. The positive–highest-G negative margin increases by 0.147 on NQ (95% interval: 0.143–0.152) and 0.024 on MS MARCO (0.015–0.033).

Two-Dimensional Illustration. Within each dataset, we fit one PCA basis jointly to the normalized query and selected SID vectors before and after adaptation, using shared axis ranges. The first two components explain 23.5% of the joint variance on NQ and 30.4% on MS MARCO. Background points show a fixed sample of up to 450 eligible queries and their selected candidates per dataset; three foreground examples are fixed by a query-ID hash, independently of their score changes. These projections illustrate the space, while all cosine comparisons use the original 768-dimensional vectors.

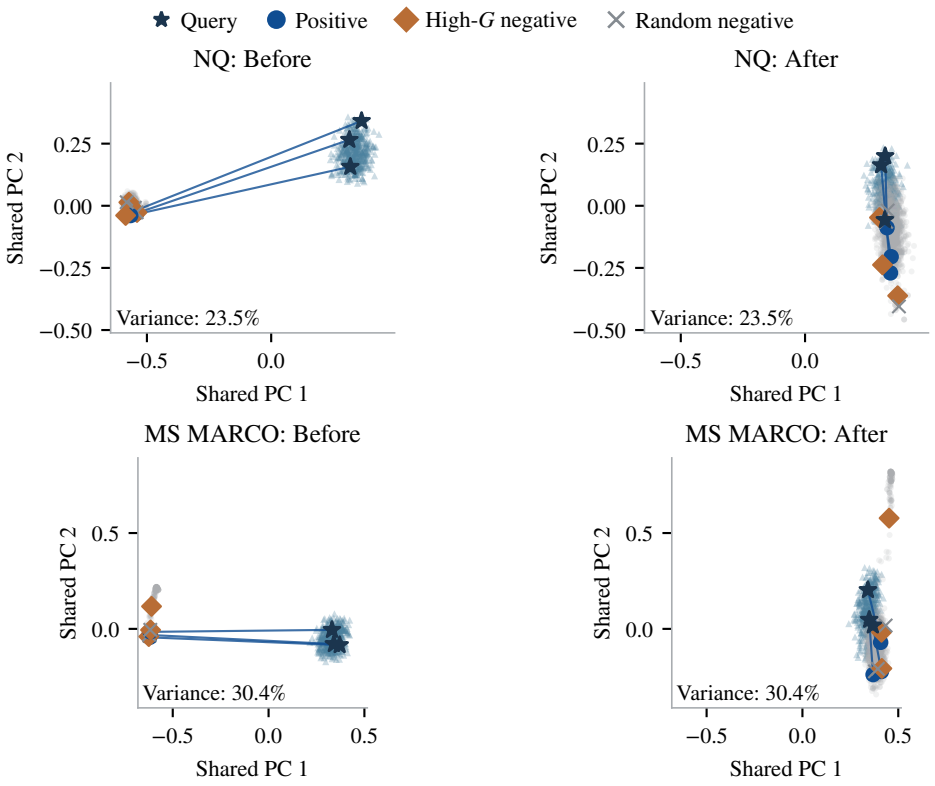


Figure 8: Shared PCA views before and after adaptation. Lines connect three fixed queries to their positive SIDs; foreground markers also identify their high-G and random negative candidates. The same basis and axis ranges are used within each dataset.

C Qualitative Retrieval Examples

Twelve examples cover both datasets and identifier families using the frozen main-method heads. G, R, and $F = G + \beta R$ rank the same generated candidates. Green panels mark annotated relevant documents. Corpus excerpts use normalized punctuation and shortened titles or headings. > 100 denotes ranks beyond the reported cutoff.

C.1 NQ320K with PQ identifiers

Example C.1 Distinguishing neighboring winter sports

Query: what's the difference between downhill and super g

Top under G

Snowboarding at the Winter Olympics

Snowboarding is a sport at the Winter Olympic Games.

Annotated relevant document

Super-G

The course is set so that skiers must turn more than in downhill

Document rank	G	R	F
Annotated relevant	45	1	**1**
Alternative	1	31	12

Observation. Matching promotes the annotated Super-G page from rank 45 to rank 1, separating the requested skiing event from snowboarding.

Example C.2 Identifying natural pollution sources

Query: where do some of the natural air pollutants come from

Top under G

Indoor air pollution in developing nations

rely on biomass, in the form of wood, charcoal, dung, and crop residue

Annotated relevant document

Air pollution

Volcanic activity, which produces sulfur, chlorine, and ash particulates

Document rank	G	R	F
Annotated relevant	3	3	**1**
Alternative	1	17	4

Observation. Neither score alone places the annotated page first. Their combination promotes the page describing natural pollution sources to rank 1.

Example C.3 Preserving the requested highway identity

Query: where does us highway 1 start and end

Top under R

U.S. Route 231

U.S. Route 231 is a parallel route of U.S. Route 31.

Annotated relevant document

U.S. Route 1

from Fort Kent, Maine, at the Canada – US border, south to Key West, Florida

Document rank	G	R	F
Annotated relevant	1	21	**1**
Alternative	62	1	40

Observation. Matching alone favors a different numbered highway; fusion preserves the generation preference for the annotated U.S. Route 1 page.

C.2 NQ320K WITH TITLE–URL IDENTIFIERS

Example C.4 Resolving a song and its artist

Query: who wrote the song photograph by ringo starr

Top under G
My Generation: The Very Best of The Who
one of The Who's many greatest hits collections

Annotated relevant document
Photograph (Ringo Starr song)
Starr co-wrote the song with George Harrison, his former bandmate from the Beatles.

Document rank	G	R	F
Annotated relevant	19	1	**1**
Alternative	1	> 100	> 100

Observation. Matching promotes the annotated Photograph page from rank 19 to rank 1; fusion retains the song-specific result.

Example C.5 Combining two second-place signals

Query: derived s.i. unit of joule (j) in terms of fundamental units

Top under G
Units of energy
the SI unit for energy is the same as the unit of work

Annotated relevant document
Joule
is a derived unit of energy in the International System of Units.

Document rank	G	R	F
Annotated relevant	2	2	**1**
Alternative	1	6	2

Observation. Both individual scores place the annotated Joule page second; their combination ranks the specific unit page first.

Example C.6 Retaining the geophysical interpretation

Query: paleomagnetism is a dating method based on changes in the earth's polarity as demonstrated by

Top under R
Rate of evolution
The rate of evolution is a variable of considerable interest in evolutionary biology.

Annotated relevant document
Paleomagnetism
Magnetic stripes are the result of reversals of the Earth's field and seafloor spreading.

Document rank	G	R	F
Annotated relevant	1	7	**1**
Alternative	> 100	1	4

Observation. Fusion retains the paleomagnetism page preferred by generation, while matching alone favors an article about evolutionary biology.

C.3 MS MARCO WITH PQ IDENTIFIERS

Example C.7 Retrieving the named deity

Query: which greek deity was honored with a towering gold-and-ivory statue inside the parthenon?

Top under G
Reconstruction of the Parthenon
the building of the Parthenon was Pericles' greatest triumph

Annotated relevant document
The Glorious Parthenon
a 40-foot, gold-and-ivory statue of the goddess Athena.

Document rank	G	R	F
Annotated relevant	3	1	**1**
Alternative	1	6	4

Observation. Matching promotes the annotated document explicitly naming Athena above the construction-focused alternative.

Example C.8 Complementary rankings for a chemistry query

Query: what does oxygen and hydrogen make

Top under G
How is it a liquid?
how these molecules (made up of atoms) interact with each other.

Annotated relevant document
Hydrogen + Oxygen = Water
two molecules of the diatomic hydrogen gas, combined with one molecule of the diatomic oxygen gas

Document rank	G	R	F
Annotated relevant	4	3	**1**
Alternative	1	9	2

Observation. Fusion moves the annotated chemistry document to rank 1; generation and matching separately place it fourth and third.

Example C.9 Keeping adjacent military ranks distinct

Query: what is a colonel in the air force

Top under R
Lieutenant colonel
above a major and below a colonel.

Annotated relevant document
Air Force Colonel
Colonel (Col) is the third Field Officer rank in the United States Air Force.

Document rank	G	R	F
Annotated relevant	1	5	**1**
Alternative	3	1	3

Observation. Generation distinguishes colonel from lieutenant colonel; fusion retains its annotated rank-one result despite the matching-only preference.

C.4 MS MARCO WITH TITLE–URL IDENTIFIERS

Example C.10 Recognizing the hospital-management role

Query: what does a coo do at a hospital

Top under G

Occupational health advice

Fit for Work offers free, expert and impartial advice to anyone looking for help

Annotated relevant document

Hospital Chief Operating Officer

responsible for creating an efficient management structure within the hospital.

Document rank	G	R	F
Annotated relevant	7	1	**1**
Alternative	1	65	11

Observation. Matching promotes the annotated hospital COO document from rank 7 to rank 1, ahead of the occupational-health alternative.

Example C.11 Finding the cause of plate movement

Query: what is the cause for tectonic plates

Top under G

Compressional boundary

the plates are moving towards each other.

Annotated relevant document

What Causes Tectonic Plates to Move?

plate movement originates from convection currents in the mantle.

Document rank	G	R	F
Annotated relevant	5	6	**1**
Alternative	1	34	4

Observation. Fusion promotes the annotated explanation of plate movement, although generation and matching independently rank it fifth and sixth.

Example C.12 Preserving a definition over a usage guide

Query: what is a dry chemical fire extinguisher

Top under R

How to Use a Fire Extinguisher

Determine a safe evacuation path before approaching the fire.

Annotated relevant document

Types of Fire Extinguishers

Dry Chemical fire extinguishers extinguish the fire primarily by interrupting the chemical reaction of the fire triangle.

Document rank	G	R	F
Annotated relevant	1	8	**1**
Alternative	23	1	9

Observation. Fusion keeps the annotated description of dry-chemical extinguishers at rank 1; matching alone favors a general usage guide.