Learning When to Think: Utility-Guided Adaptive Reasoning for Efficient Text Retrieval

Yingtao Luo[1] **Kangyi Zhao**[2] **Shiyi Du**[1] **Yubo Li**[1]
[1]Carnegie Mellon University [2]University of Pittsburgh
{yingtaol, shiyid, yubol}@andrew.cmu.edu
kaz78@pitt.edu

Abstract

Reasoning-conditioned embeddings can improve retrieval by progressively refining semantic constraints. However, additional reasoning may introduce over-generalized or semantically noisy abstractions that weaken retrieval specificity. Existing methods largely supervise reasoning through final retrieval outcomes, providing limited guidance over the refinement process itself. We propose AURA, a framework that directly supervises intermediate reasoning states according to how well they separate relevant documents from hard distractors. AURA further trains a lightweight utility estimator over prefix-conditioned embeddings to adapt reasoning depth at inference time. On the reasoning-intensive BRIGHT benchmark, AURA-4B achieves 34.7 average nDCG@10 while using 2.8 reasoning steps and 96 generated tokens per query. Under matched deployment conditions, adaptive stopping reduces latency from 4.6 to 0.76 seconds relative to full-trajectory reasoning. Results on MS MARCO further suggest that this specialization need not substantially degrade conventional in-domain retrieval.

1 Introduction

Recent dense retrieval systems increasingly incorporate reasoning through query rewriting, decomposition, reranking, and reasoning-conditioned embeddings (Qin et al., 2025; Jiang et al., 2025; Asai et al., 2024; Li et al., 2025b; Muennighoff et al., 2024; Tang et al., 2025; Lan et al., 2025), motivated by the observation that semantic refinement can improve retrieval for complex queries.

However, existing reasoning-based retrieval systems generally treat additional reasoning as beneficial, applying it without distinguishing whether further refinement helps or hurts retrieval quality. In practice, retrieval-oriented reasoning often exhibits diminishing returns. While some refinements sharpen *retrieval specificity*—the separation between relevant documents and hard distractors—by identifying useful constraints or disambiguating entities, others introduce semantic drift or overly abstract reformulations that weaken retrieval discrimination (Fan et al., 2026; Kotte, 2026). As a result, longer reasoning trajectories do not always improve retrieval quality, and can even harm retrieval discrimination, increase inference latency, and introduce unnecessary computational cost.

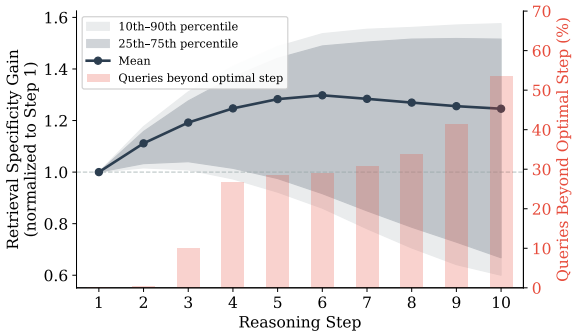

Figure 1: Retrieval specificity across reasoning steps on BRIGHT. Lines and shaded regions show the mean and query-level variability normalized to Step 1. Bars indicate the fraction of queries that have already passed their query-specific optimal step.

To understand how retrieval utility evolves during reasoning, we analyze retrieval specificity across intermediate reasoning steps on BRIGHT. As shown in Figure 1, retrieval specificity improves on average during early reasoning steps, suggesting that additional refinement can help construct more discriminative retrieval representations. However, this improvement is highly heterogeneous: variability increases rapidly with reasoning depth, indicating that different queries benefit from substantially different amounts of reasoning. The fraction of queries that have already passed their optimal reasoning step increases and exceeds 50% by Step 10. This evidence supports the notion that additional reasoning can be unnecessary and may even reduce

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 34424–34434
October 24-29, 2026 ©2026 Association for Computational Linguistics

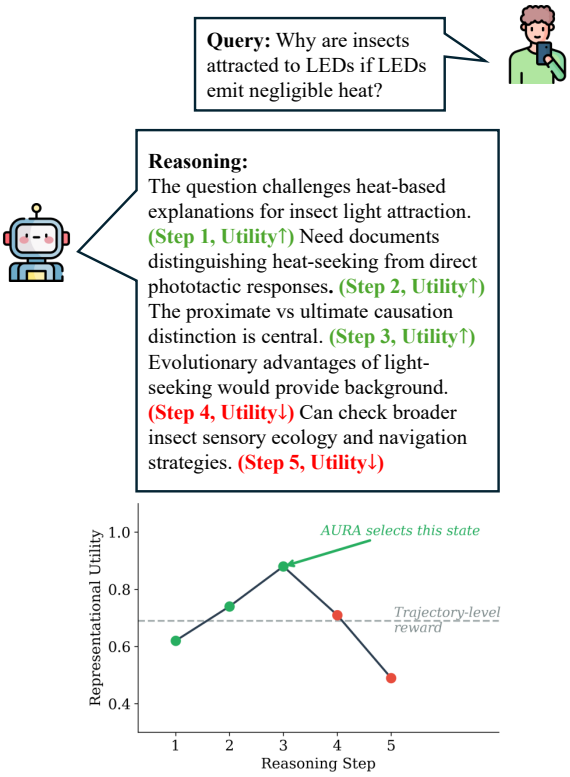

Figure 2: Motivation behind AURA. Existing methods supervise only the final retrieval outcome, while AURA identifies the peak-utility reasoning state before over-reasoning weakens retrieval specificity.

the retrieval quality of many queries. Effective retrieval requires adapting the depth of the reasoning to the query-specific need.

A key limitation of existing methods is that reasoning is typically supervised only through final retrieval performance. While effective for evaluating end-to-end success, these coarse signals provide little guidance over the reasoning process. More importantly, current retrieval systems cannot predict whether additional reasoning will improve retrieval before observing retrieval outcomes, and thus can only allocate computation reactively: they expand reasoning trajectories regardless of whether further refinement is likely to help.

In this work, we formulate retrieval-oriented reasoning as a utility-guided computation allocation problem. Rather than treating reasoning as an unconditional token expansion, we evaluate whether each intermediate refinement improves the model's ability to distinguish relevant documents from challenging distractors (Figure 2). Under this perspective, useful reasoning is determined not by trajectory length or textual sophistication, but by its measurable effect on retrieval discrimination.

Building on this perspective, we propose **AURA**[1] (**A**daptive **U**tility-aware **R**easoning Retriev**A**l), a framework for learning retrieval-oriented reasoning through utility supervision over intermediate reasoning states. During training, AURA uses relevance labels and retrieval candidates to estimate whether each refinement step improves retrieval discrimination, encouraging reasoning trajectories that sharpen retrieval specificity while avoiding semantically noisy refinements.

Since retrieval specificity is unavailable at inference time, AURA additionally trains a lightweight utility estimator to predict the retrieval utility of each current reasoning state. Monitoring this estimate online allows AURA to stop when state utility ceases to improve, without exhaustively expanding the reasoning trajectory.

We evaluate AURA on the 12 reasoning-intensive retrieval domains in BRIGHT (Hongjin et al., 2025) and on MS MARCO (Bajaj et al., 2016) as a conventional in-domain retrieval setting. AURA outperforms the compared reasoning-conditioned retrieval methods on BRIGHT. On MS MARCO, it remains comparable to a strong reasoning-trained retriever, suggesting no substantial in-domain degradation when extensive reasoning is unnecessary. Controlled analyses further show that the gains cannot be explained by denser intermediate contrastive supervision or trajectory filtering alone, and that retrieval specificity is strongly aligned with corpus-level nDCG@10.

Our contributions are summarized as follows:

- We formulate reasoning-conditioned retrieval as an adaptive refinement process and show that reasoning utility is highly heterogeneous.
- We propose **AURA**, a framework that supervises intermediate reasoning states according to their retrieval specificity, and trains a lightweight utility estimator to dynamically allocate reasoning during inference.
- We demonstrate improvements over the compared methods across 12 reasoning-intensive domains, together with a substantially better performance–cost trade-off under matched inference conditions.

[1]Project repository: https://github.com/yingtaoluo/Adaptive-Reasoning-Retrieval.

2 Related Work

2.1 Dense Retrieval and Embedding Models

Modern dense retrieval commonly relies on bi-encoder architectures that map queries and documents into a shared embedding space, where retrieval is performed through vector similarity (Karpukhin et al., 2020). Early embedding approaches such as Sentence-BERT (Reimers and Gurevych, 2019) demonstrated the effectiveness of contrastive representation learning for semantic retrieval, followed by stronger instruction-tuned and large embedding models including INSTRUCTOR (Su et al., 2023), E5 (Wang et al., 2022), and recent LLM-based embedding systems (Muennighoff et al., 2024; Liu et al., 2024; Wang et al., 2024). Recent studies further explore using LLM hidden states directly as embeddings (Ma et al., 2024; Lee et al., 2025) or leveraging in-context learning to improve representation quality (Li et al., 2025a). Beyond supervised retrieval data, dense retrievers have also been trained using unsupervised contrastive objectives and synthetic supervision (Lee et al., 2019; Izacard et al., 2021; Chen et al., 2024; Ji et al., 2025; Yoon et al., 2025). Despite strong performance, most embedding models remain fundamentally single-pass encoders that construct representations without explicit semantic refinement or intermediate reasoning.

2.2 Reasoning for Retrieval

Recent work increasingly incorporates reasoning into retrieval systems through query rewriting, decomposition, iterative retrieval, and reasoning-augmented reranking (Qin et al., 2025; Jiang et al., 2025; Asai et al., 2024; Li et al., 2025b; Long et al., 2025). For example, ReasonIR (Shao et al., 2025) synthesizes reasoning trajectories and hard negatives to train retrieval models on reasoning-intensive benchmarks (Hongjin et al., 2025). More recently, reasoning has been incorporated directly into embedding formation: multimodal methods such as RITE (Liu et al., 2025b), LREM (Tang et al., 2025), and UME-R1 (Lan et al., 2025) generate reasoning before constructing retrieval representations, with further gains from inference-time scaling through sampling multiple reasoning trajectories (Muennighoff et al., 2024).

However, reasoning is not uniformly beneficial for retrieval. Query expansion has long been known to suffer from query drift (Mitra et al., 1998; Carpineto and Romano, 2012), and recent work shows that LLM-based reasoning can hurt dense retrieval due to semantic drift (Fan et al., 2026; Kotte, 2026). Similar observations have emerged for chain-of-thought prompting more broadly (Liu et al., 2025a; Hassid et al., 2025; Sui et al.). Despite these findings, existing methods either supervise reasoning only through final retrieval outcomes or select among complete trajectories via best-of-k, providing no direct supervision over intermediate refinement states nor adaptive reasoning allocation at inference time. AURA addresses both gaps.

3 Methodology

3.1 Unified Reasoning Embedding

To ground reasoning-conditioned retrieval within a unified framework, AURA utilizes an autoregressive language model as its unified backbone f_θ. To enable a single language model to simultaneously optimize autoregressive token generation and extract high-quality dense retrieval representations, we propose a structured sequence format. Given a query x, the format is as follows:

Unified Sequence Format:
`[User]`: Input: x. Generate step-by-step reasoning.
`[Assistant]`: `<think>` $r_1, r_2, \ldots, r_T$ `</think>`

where $r = (r_1, \ldots, r_T)$ represents the generated reasoning trajectory parsed at sentence boundaries.

Embedding extraction. At each sentence boundary t, we construct a prefix-conditioned embedding by passing the query and reasoning prefix through the backbone and extracting the final-layer hidden state at the last token position of r_t:

$$z_t = \mathrm{proj}(\mathrm{LastHidden}_\theta(x, r_{\leq t})), \qquad (1)$$

where $\mathrm{LastHidden}_\theta(\cdot)$ returns the final layer's hidden state at the last token of the input, and $\mathrm{proj}(\cdot)$ is a linear projection to the embedding space.

Retrieval Specificity Scoring. To supervise intermediate reasoning states, AURA defines a retrieval specificity score that measures how well a given reasoning state separates relevant documents from distractors. For a training query x, let d^+ denote a relevant document and let $\mathcal{D}^-(x)$ denote the set of in-batch negatives. Given z_t, we define its retrieval specificity score as

$$S(z_t) = \mathrm{sim}(z_t, z_{d^+}) - \frac{1}{k}\sum_{i=1}^{k} \mathrm{sim}(z_t, z_{d_i^-}), \quad (2)$$

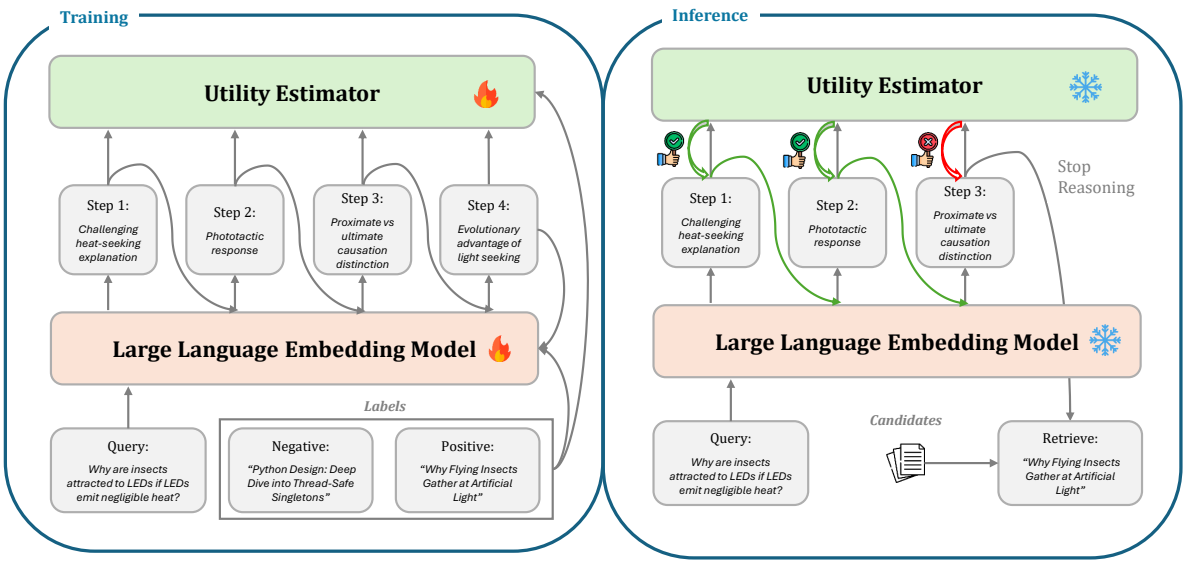

Figure 3: Overview of AURA. **Training:** A warm-up stage first establishes a retrieval-oriented embedding space. Specificity labels are then constructed from the warmed-up model, the utility estimator is trained on frozen embeddings, and utility-aware weighting is enabled for subsequent joint optimization. **Inference:** The utility estimator evaluates each state online and terminates reasoning according to the predicted utility, returning the selected embedding for retrieval.

where $d_1^-, \ldots, d_k^-$ are the top-k highest-similarity distractors from $\mathcal{D}^-(x)$ according to $\mathrm{sim}(\cdot,\cdot)$ as cosine similarity. $S(z_t)$ evaluates the retrieval specificity at each intermediate reasoning state. The marginal effect of a reasoning step can then be measured as $\Delta S_t = S(z_t) - S(z_{t-1})$.

Training objectives. The model is trained with a multi-task loss combining contrastive retrieval and language modeling objectives:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{InfoNCE}} + \mathcal{L}_{\text{CE}} \tag{3}$$

The autoregressive objective is computed over all reasoning tokens, preserving the model's ability to generate coherent reasoning:

$$\mathcal{L}_{\text{CE}} = -\sum_{t=1}^{T} \log P_\theta(r_t \mid x, r_{<t}) \tag{4}$$

The contrastive objective optimizes the prefix-conditioned embeddings to separate relevant documents from distractors. Unlike existing methods that apply InfoNCE only at the final state z_T, AURA optimizes at all intermediate states $z_1, \ldots, z_T$. For each intermediate embedding z_t, we compute the InfoNCE loss against the relevant document embedding z_{d^+} and in-batch negatives:

$$\mathcal{L}_{\text{InfoNCE}} = -\frac{1}{T}\sum_{t=1}^{T} w_t \log \frac{\exp(z_t \cdot z_{d^+}/\tau)}{\sum_{d\in\mathcal{D}} \exp(z_t \cdot z_d/\tau)} \tag{5}$$

where $\mathcal{D} = \{d^+\} \cup \{d_j^-\}$ contains the relevant document and all in-batch negatives, τ is the contrastive temperature (default set to 0.02), and the weight is set as

$$w_t = \frac{\exp(\max(0, \mathrm{stopgrad}(\Delta S_t)))}{\sum_{t'} \exp(\max(0, \mathrm{stopgrad}(\Delta S_{t'})))}, \tag{6}$$

which emphasizes the learning of reasoning refinements that produce larger gains in retrieval specificity and reduces supervision on saturated or retrieval-harmful reasoning states. Here, we apply stop-gradient so that utility scores are used only to reweight contrastive updates, preventing the model from artificially increasing utility values instead of improving retrieval representations.

Gradient isolation. Since both objectives share the same backbone model and operate on overlapping reasoning positions, naive joint optimization may cause contrastive gradients to interfere with autoregressive generation. To mitigate this effect, during the contrastive optimization, we apply stop-gradient to all positions preceding the embedding extraction point, such that gradients from $\mathcal{L}_{\text{InfoNCE}}$ do not propagate through $(x, r_{<t})$.

Under this design, $\mathcal{L}_{\text{CE}}$ retains full gradient flow across the reasoning sequence, while $\mathcal{L}_{\text{InfoNCE}}$ only updates representations at the current reasoning step without backpropagating through earlier

reasoning prefixes. This separation reduces optimization interference between generation and retrieval objectives, allowing the model to learn discriminative retrieval representations while preserving autoregressive reasoning quality.

3.2 Inference-Time Utility Estimation

Since retrieval specificity $S(z_t)$ depends on relevance labels and distractors, it is unavailable at inference time. To enable adaptive reasoning, AURA learns a utility estimator that predicts the retrieval specificity of the current intermediate state. For each state z_t, we define its target utility as

$$y_t = S(z_t), \tag{7}$$

which represents the observed state-level retrieval utility. The estimator

$$V_\phi(z_t) \tag{8}$$

is trained to preserve the ordering of state utility using a pairwise ranking loss:

$$\mathcal{L}_{\text{rank}} = -\log \sigma(V_\phi(z_i) - V_\phi(z_j)), \qquad y_i > y_j. \tag{9}$$

Since adaptive stopping and state selection depend only on the relative ordering of reasoning states within a query trajectory, this objective targets ordinal agreement rather than absolute score calibration and is less sensitive to query- and negative-set-specific shifts in the scale of retrieval specificity. At inference time, reasoning terminates when the predicted state utility has not improved for p consecutive steps, and the generated state with the highest predicted utility is used for retrieval. The utility estimator is a lightweight MLP over prefix-conditioned embeddings.

4 Experiments

4.1 Training Data Construction

We build on the publicly released training data with 101K query-positive-negative samples from ReasonIR (Shao et al., 2025), which provides reasoning-intensive queries with paired positive documents and hard negatives synthesized from MS MARCO[2] seed documents. To obtain reasoning trajectories for utility supervision, we sample G=8 independent trajectories per query from DeepSeek-R1 (Guo et al., 2025), each truncated to a maximum of l=2,048 tokens. Each trajectory is segmented at sentence boundaries into intermediate reasoning states. After the retrieval warm-up described below, we compute retrieval specificity $S(z_t)$ for every intermediate state using the labeled positive document and hard negatives provided by ReasonIR. Trajectories where no intermediate state improves over the initial query embedding are removed during label construction, retaining 67.5% of sampled trajectories (approximately 5.4 per query on average), for approximately 545K reasoning samples.

4.2 Training Procedure

Training separates label construction from utility-aware optimization. First, we warm up the unified model for one epoch with the multi-task objective but without utility-guided weighting. We freeze this warmed-up checkpoint to construct retrieval-specificity and state-utility labels. The utility estimator V_ϕ, a 3-layer 10M-parameter MLP with ReLU activations, is trained on these frozen embeddings with a per-GPU batch size of 256 and a learning rate of 5×10^{-4} for up to ten epochs, with early stopping based on loss convergence. We then enable utility-aware weighting and continue joint optimization of the unified model. AdamW is used throughout. This schedule avoids constructing labels from the backbone's initial, non-retrieval-adapted representations.

4.3 Implementation Details

Experiments are conducted on 8 NVIDIA H100 GPUs, consuming approximately 100 GPU hours to converge. We adopt Qwen3-4B-Instruct as the backbone model and use a per-GPU batch size of 128 and a learning rate of 1×10^{-5}. The embedding dimension matches the backbone hidden size. We primarily report nDCG@10, together with generated tokens, average reasoning steps, and matched-setting wall-clock latency.

4.4 Baselines

We organize baselines into three categories according to whether inference-time reasoning is used and how reasoning is integrated with retrieval.

Setting I evaluates standard retrieval systems without explicit inference-time reasoning, where representations are constructed directly from the original query. Setting II evaluates retrieval augmented with external query rewriting, which follows the decoupled paradigm of recent reasoning-

[2] https://huggingface.co/datasets/microsoft/ms_marco

Table 1: nDCG@10 on BRIGHT under three inference settings.

Method	Avg.	StackExchange							Coding		Theorem-based		
		Bio.	Earth.	Econ.	Psy.	Rob.	Stack.	Sus.	Leet.	Pony	AoPS	TheoQ.	TheoT.
Setting I: Single-pass retrieval with original query													
BM25 (Robertson et al., 2009)	14.5	18.9	27.2	14.9	12.5	13.6	18.4	15.0	24.4	7.9	6.2	10.4	4.9
SBERT (Reimers and Gurevych, 2019)	14.9	15.1	20.4	16.6	22.7	8.2	11.0	15.3	26.4	7.0	5.3	20.0	10.8
Qwen3-4B (Zhang et al., 2025)	5.6	3.5	8.0	2.3	2.0	1.6	1.0	4.4	2.1	0.1	4.9	18.0	19.2
OpenAI text-embedding-3-large	17.9	23.3	26.7	19.5	27.6	12.8	14.3	20.5	23.6	2.4	8.5	23.5	11.7
ReasonIR-8B (Shao et al., 2025)	24.4	26.2	31.4	**23.3**	30.0	18.0	**23.9**	20.5	35.0	10.5	**14.7**	31.9	27.2
RaDeR-7B (Das et al., 2025)	25.5	34.6	38.9	22.1	33.0	14.8	22.5	23.7	37.3	5.0	10.2	28.4	35.1
DIVER-4B (Long et al., 2025)	**28.9**	**41.8**	**43.7**	21.7	**35.3**	**21.0**	21.2	**25.1**	**37.6**	**13.2**	10.7	**38.4**	**37.3**
Setting II: Retrieval with external GPT-generated query expansion													
BM25	27.0	**53.6**	**54.1**	24.3	38.7	18.9	**27.7**	26.3	19.3	17.6	3.9	19.2	20.8
SBERT	17.8	18.5	26.3	17.5	27.2	8.8	11.8	17.5	24.3	10.3	5.0	22.3	23.5
Qwen3-4B	5.5	1.3	17.3	2.5	6.2	1.0	4.8	4.5	3.0	5.9	0.0	7.2	12.5
OpenAI text-embedding-3-large	23.3	35.2	40.1	25.1	38.0	13.6	18.2	24.2	24.5	6.5	7.7	23.8	22.9
ReasonIR-8B	29.9	43.6	42.9	**32.7**	38.8	20.9	25.8	**27.5**	31.5	**19.6**	7.4	33.1	35.7
RaDeR-7B	29.2	36.1	42.9	25.2	37.9	16.6	27.4	25.0	**34.8**	11.9	**12.0**	37.7	**43.4**
DIVER-Retriever-4B	**32.1**	51.9	53.5	29.5	**41.2**	**21.4**	27.5	26.1	33.5	11.7	9.5	**39.3**	39.7
Setting III: Retrieval with adaptive internal reasoning (w/o external LLM)													
AURA-4B (base)	**34.7**	**47.5**	**48.6**	**28.3**	**43.5**	**23.8**	**31.2**	**34.6**	**40.9**	**14.2**	**17.0**	**43.2**	**43.3**

Table 2: Effect of Qwen3-4B external query rewriting on BRIGHT (average nDCG@10).

Retriever	**Original**	**+ Rewrite**
BM25	14.5	12.9
SBERT	14.9	12.6
Qwen3-4B	5.6	5.5
OpenAI-3-large	17.9	15.2

intensive retrieval works, where reasoning is generated by an external LLM at inference time.

To provide strong baselines, we intentionally adopt GPT-4 for query expansion rather than external reasoners with comparable size to our Qwen3-4B backbone. We also evaluate Qwen3-4B-Instruct, AURA's backbone, as an external query rewriter using a prompt comparable to the GPT-4 baseline. As shown in Table 2, it does not improve any evaluated retriever and reduces average BRIGHT nDCG@10 by 0.1–2.7 points. We therefore use GPT-4 rewriting to construct a stronger external-reasoning baseline.

Setting III evaluates endogenous reasoning, the new paradigm of AURA, where reasoning and embedding formation are performed within the same model. Our method further introduces utility estimation to adaptively allocate inference-time reasoning and avoid unnecessary refinement.

4.5 Main Results

Table 1 reports retrieval performance of different models measured by nDCG@10 on BRIGHT benchmark (Hongjin et al., 2025).

First, Setting II consistently outperforms Setting I across retrievers. This result is aligned with recent reasoning-intensive retrieval literature and suggests that augmenting retrieval with external query rewriting can substantially improve retrieval quality. Notably, Setting II baselines use a separate external reasoning model (GPT-4) for query expansion while keeping the retriever unchanged. Across methods, this external reasoning strategy appears to provide the largest improvements for lexical retrievers such as BM25.

AURA achieves the best overall performance among the compared systems with an average nDCG@10 of 34.7. Despite not relying on any external reasoning model, AURA outperforms all Setting II baselines that are powered by GPT-4 query expansion. The gains are especially pronounced in the theorem-based subsets, where AURA shows the largest margin over existing approaches. The strongest external reasoning baseline, DIVER-Retriever-4B, achieves 32.1 with GPT-4 expanded queries, while AURA improves this to 34.7 (+2.6) using reasoning generated entirely within the retrieval model itself.

Compared with Setting I, the advantage of adaptive reasoning is even larger. The strongest baseline without inference-time reasoning achieves 28.9, whereas AURA reaches 34.7 (+5.8 absolute, +20.1% relative).

4.6 Alignment with Corpus-Level Retrieval

Retrieval specificity is computed from labeled positives and hard negatives during training, whereas the final evaluation ranks the full corpus. To test whether the local signal reflects the listwise objective, we perform full-corpus BRIGHT retrieval

for every available intermediate state in each naturally varying-length reasoning trajectory. For each query, we compute Spearman correlation across its states between retrieval specificity and corpus-level nDCG@10, and then average the query-level correlations. The resulting average correlation is 0.97. Moreover, the state selected by oracle specificity-based selection is also nDCG@10-optimal for 98.1% of queries. This close agreement indicates that retrieval specificity tracks corpus-level ranking quality rather than only in-batch discrimination.

4.7 Retrieval on Traditional IR Benchmarks

Table 3 presents results on MS MARCO passage retrieval (Bajaj et al., 2016). We evaluate on the official development set, TREC-DL'19, and TREC-DL'20, all of which use held-out queries disjoint from the training split. These splits provide an in-domain check outside the reasoning-intensive BRIGHT setting. AURA-4B is comparable to RaDeR-7B across all three splits despite using a smaller backbone. This result suggests that utility-aware reasoning supervision does not substantially degrade conventional in-domain retrieval relative to a strong reasoning-trained retriever; broader general-purpose evaluation is left to future work.

4.8 Ablation Study

Table 4 reports ablation results on BRIGHT. We examine five design dimensions of AURA, each targeting a specific question about the framework.

Q1: Is the specificity score robust to negative aggregation? The choice of negative aggregation in $S(z_t)$ has minimal impact: mean top-3, max, and mean all negatives yield similar performance (34.7, 34.6, 34.4). This indicates that AURA is robust to this design choice. Mean over all negatives may dilute the signal from hard negatives, while using only the hardest single negative may be susceptible to outlier noise; the default mean top-3 offers a reasonable balance.

Q2: Does utility-aware supervision improve over dense intermediate supervision? This is the central ablation: we ask whether the retrieval specificity signal provides meaningful supervision beyond simply applying contrastive training at more positions. We compare four different supervision strategies: (1) *final state only*, which applies InfoNCE only at z_T as in prior work; (2) *random utility labels*, which trains the utility estimator with randomly assigned labels; (3) *monotonic utility labels*, which assumes reasoning quality increases monotonically with depth; and (4) *uniform-weight intermediate states*, which applies InfoNCE at all intermediate states with equal weight, removing utility-based differentiation.

Results show a clear ordering: final-state-only (31.2) $<$ random (31.9) $<$ monotonic (33.2) $<$ uniform (33.6) $<$ AURA (34.7). Two findings are notable. First, uniform-weight supervision already improves substantially over final-state-only (+2.4), confirming that denser contrastive training is beneficial. However, AURA outperforms uniform weighting by a further 1.1 points, demonstrating that utility-aware differentiation among intermediate states provides supervision that cannot be achieved by simply increasing the number of contrastive training positions. Second, monotonic labels underperform uniform weighting (-0.4), indicating that the assumption of monotonically increasing reasoning quality is actively harmful—consistent with the over-reasoning phenomenon motivating this work. Random labels degrade close to final-state-only, confirming that the pairwise ranking loss requires accurate ordering signal to function effectively.

Q3: Are trajectory filtering and gradient isolation necessary? Retaining trajectories in which no intermediate state improves over the initial query changes nDCG@10 only marginally, from 34.7 to 34.6. Filtering therefore provides a small benefit by removing trajectories without a positive within-trajectory ranking signal, but does not explain AURA's overall improvement. In contrast, removing gradient isolation causes a substantial drop to 29.8. Retrieval gradients propagated through earlier prefixes therefore interfere with stable joint generation–retrieval optimization, supporting the gradient-isolation design.

Q4: How well does the utility estimator approximate oracle selection? At inference time, ground-truth retrieval specificity is unavailable, which motivates utility estimation. The utility estimator V_ϕ achieves 34.7, 0.3 points below the oracle upper bound (35.0), where the oracle selects the reasoning state with the highest true $S(z_t)$ using ground-truth relevance labels. Random state selection drops to 32.3. This narrow aggregate performance gap indicates that V_ϕ provides an effective approximation of retrieval utility without access to relevance labels.

Table 3: Performance on MS MARCO passage retrieval (DEV and TREC DL19/DL20) as a standard retrieval setting rather than a reasoning-intensive one. Baseline results are from (Das et al., 2025).

Model	DEV		DL19	DL20
	MRR@10	R@1k	nDCG@10	nDCG@10
BM25 (Robertson et al., 2009)	18.4	85.3	50.6	48.0
ANCE (Xiong et al., 2020)	33.0	95.9	64.5	64.6
CoCondenser (Gao and Callan, 2022)	38.2	98.4	71.7	68.4
OpenAI Ada2	34.4	98.6	70.4	67.6
SimLM (Wang et al., 2023)	41.1	98.7	71.4	69.7
RepLLaMA (Ma et al., 2024)	**41.2**	**99.4**	**74.3**	**72.1**
RaDeR-7B (Das et al., 2025)	34.4	98.1	71.2	70.7
AURA-4B (base)	34.1	98.0	71.2	70.4

Table 4: Ablation study evaluated by average nDCG@10 on BRIGHT benchmark.

Variant	nDCG@10
AURA (base)	34.7
Specificity scoring	
Mean top-3 negatives (default)	34.7
max negative	34.6
Mean all negatives	34.4
Training supervision	
All intermediate states (default)	34.7
Final state only (z_T)	31.2
Uniform-weight intermediate states	33.6
Monotonic utility labels	33.2
Random utility labels	31.9
Data construction and optimization	
Retain all trajectories	34.6
Without gradient isolation	29.8
Inference-time utility estimation	
Utility estimator (default)	34.7
LLM-confidence stopping (3.2 steps)	32.8
Random selection	32.3
Oracle specificity-based selection	35.0
Utility-aware stopping criterion	
V_ϕ with patience p=1 (default)	34.7
Greedy peak stopping (p=0)	34.6
V_ϕ with patience p=3	33.7
No early stopping (full trajectory)	33.0

Q5: Are simple adaptive stopping rules sufficient? We compare AURA with an LLM-confidence rule based on the mean token entropy of the current reasoning step, tuning both the threshold and stopping direction on development data. Its strongest variant obtains 32.8 nDCG@10 at 3.2 average steps, compared with AURA's 34.7 at 2.8 steps. Thus, generation confidence alone does not reliably identify the reasoning state that is most discriminative for retrieval.

Q6: When should reasoning stop? Early stopping consistently outperforms using the full reasoning trajectory (33.0), validating that over-reasoning degrades retrieval quality. Among stopping strategies, patience p=1 (34.7) performs best; greedy peak stopping (p=0) is comparable (34.6), while increasing patience to p=3 (33.7) allows too many low-utility steps through. The performance gap between p=1 and no early stopping (+1.7) confirms that adaptive termination is essential.

4.9 Inference-Time Compute Efficiency

Table 5 compares inference-time reasoning cost. To isolate the effect of utility-guided stopping, we first compare AURA with and without early stopping under the same Qwen3-4B backbone, hardware, batch size, and software stack. Without early stopping, AURA generates 332 tokens (9.5 steps) per query, takes 4.6 seconds, and achieves 33.0 nDCG@10. Utility-guided stopping reduces generation to 96 tokens (2.8 steps) and latency to 0.76 seconds, corresponding to a 3.5× token reduction and approximately 6× speedup, while improving performance to 34.7 (+1.7). The utility estimator itself adds approximately 1 millisecond per query. For reference, the strongest external reasoning baseline (DIVER + GPT-4 expansion) generates 479 tokens per query using a separate reasoning model yet achieves lower performance (32.1). We do not report its wall-clock latency because external-model serving is not matched to our deployment.

Inference-time scaling through utility-guided reasoning search. Because AURA estimates retrieval utility at intermediate reasoning states, it naturally enables inference-time scaling by allocating additional reasoning computation only when further refinement is predicted to improve retrieval quality. We evaluate two search strategies under increasing inference budgets.

(1) *Best-of-k*: sample k complete reasoning trajectories independently and select the trajectory

Table 5: Inference-time reasoning compute on BRIGHT.

Method	Tokens	Steps	Latency (s)	nDCG@10
AURA (utility estimator)	**96**	**2.8**	**0.76**	**34.7**
AURA (full trajectory)	332	9.5	4.60	33.0
DIVER + GPT-4†	479	9.6	—	32.1

† Uses a separate GPT-4 model for query expansion; AURA generates reasoning within its own 4B backbone without external models.

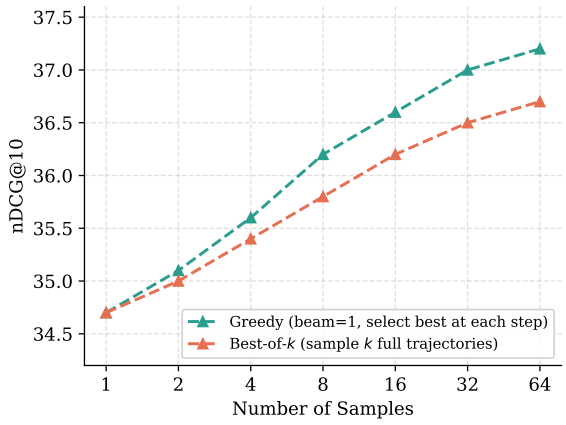

Figure 4: Inference-time scaling of AURA on BRIGHT. Greedy selects the best continuation at each step using utility estimator; best-of-k samples k full trajectories and selects the one with highest estimated utility.

with the highest estimated utility.

(2) *Greedy expansion*: generate multiple candidate refinements at each reasoning step and continue from the one with the highest predicted utility.

Figure 4 summarizes the performance-compute tradeoff on BRIGHT. Increasing search budget improves retrieval performance for both strategies, suggesting that additional reasoning can translate into stronger retrieval representations when guided by utility estimation. However, greedy expansion consistently achieves better compute efficiency than trajectory-level Best-of-k search. Performance improves from 34.7 ($k = 1$) to 37.2 ($k = 64$), compared to 36.7 under Best-of-k on the same search scale. These results suggest that retrieval-oriented reasoning benefits more from early utility-guided decisions than from selecting among completed reasoning trajectories. By prioritizing promising intermediate states throughout the reasoning process rather than evaluating trajectories only after completion, AURA allocates inference-time computation more effectively.

4.10 Qualitative Analysis

Trajectory inspection shows that discarded later steps can be semantically valid rather than incorrect. In the insects-and-LEDs example in Figure 2, early reasoning identifies the proximate-versus-ultimate causation distinction, improving separation from hard negatives. Later steps add evolutionary and sensory-ecology background. Though useful, these additions repeat encoded constraints or generalize the query toward hard negatives about evolution and insect ecology. The estimator therefore stops once the discriminative constraint is encoded. This illustrates AURA's central distinction: informative text need not provide retrieval utility; stopping should depend on document discrimination rather than textual detail.

5 Conclusion

We presented AURA, a utility-guided framework that supervises intermediate reasoning states and stops generation when predicted retrieval utility ceases to improve. AURA achieves the strongest average BRIGHT performance among the compared methods while substantially reducing reasoning cost, without an external inference-time reasoner. Controlled ablations and full-corpus analysis show that the gains arise from utility-aware supervision rather than denser supervision or trajectory filtering, and that retrieval specificity closely tracks nDCG@10. Results on MS MARCO suggest limited in-domain degradation, while utility-guided greedy expansion further improves the performance–compute trade-off.

Limitations

Despite its efficiency gains, AURA remains slower than single-pass retrieval because reasoning is generated autoregressively. Its utility labels are constructed once from a warmed-up checkpoint and may become less accurate as the embedding space evolves during subsequent optimization. Finally, our evaluation focuses primarily on reasoning-intensive English retrieval; broader validation on web search, multilingual retrieval, and generation tasks is needed to establish generality and characterize the remaining retrieval–generation trade-off.

Acknowledgments

Yingtao Luo received support from the OpenAI Researcher Access Program. Shiyi Du was partially supported by a SoftBank Group/Arm Ph.D. Fellowship. AI-assisted tools were used to improve language and presentation. The authors reviewed and take full responsibility for all technical claims and final content.

References

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, and 1 others. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Claudio Carpineto and Giovanni Romano. 2012. A survey of automatic query expansion in information retrieval. *Acm Computing Surveys (CSUR)*, 44(1):1–50.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*, 4(5).

Debrup Das, Sam O'Nuallain, and Razieh Rahimi. 2025. Rader: Reasoning-aware dense retrieval models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 19981–20008.

Yongqi Fan, Xiaoyang Chen, Dezhi Ye, Jie Liu, Haijin Liang, Jin Ma, Ben He, Yingfei Sun, and Tong Ruan. 2026. Tfrank: Think-free reasoning enables practical pointwise llm ranking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 21020–21028.

Luyu Gao and Jamie Callan. 2022. Unsupervised corpus aware language model pre-training for dense passage retrieval. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2843–2853.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Michael Hassid, Gabriel Synnaeve, Yossi Adi, and Roy Schwartz. 2025. Don't overthink it. preferring shorter thinking chains for improved llm reasoning. *arXiv preprint arXiv:2505.17813*.

SU Hongjin, Howard Yen, Mengzhou Xia, Weijia Shi, Niklas Muennighoff, Han-yu Wang, Liu Haisu, Quan Shi, Zachary S Siegel, Michael Tang, and 1 others. 2025. Bright: A realistic and challenging benchmark for reasoning-intensive retrieval. In *The Thirteenth International Conference on Learning Representations*.

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

Yifan Ji, Zhipeng Xu, Zhenghao Liu, Yukun Yan, Shi Yu, Yishan Li, Zhiyuan Liu, Yu Gu, Ge Yu, and Maosong Sun. 2025. Learning more effective representations for dense retrieval through deliberate thinking before search. *arXiv preprint arXiv:2502.12974*.

Pengcheng Jiang, Jiacheng Lin, Lang Cao, Runchu Tian, SeongKu Kang, Zifeng Wang, Jimeng Sun, and Jiawei Han. 2025. Deepretrieval: Hacking real search engines and retrievers with large language models via reinforcement learning. *Preprint*, arXiv:2503.00223.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick SH Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *EMNLP (1)*, pages 6769–6781.

Varun Kotte. 2026. Not all queries need rewriting: When prompt-only llm refinement helps and hurts dense retrieval. *arXiv preprint arXiv:2603.13301*.

Zhibin Lan, Liqiang Niu, Fandong Meng, Jie Zhou, and Jinsong Su. 2025. Ume-r1: Exploring reasoning-driven generative multimodal embeddings. *arXiv preprint arXiv:2511.00405*.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2025. NV-embed: Improved techniques for training LLMs as generalist embedding models. In *The Thirteenth International Conference on Learning Representations*.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096.

Chaofan Li, Minghao Qin, Shitao Xiao, Jianlyu Chen, Kun Luo, Defu Lian, Yingxia Shao, and Zheng Liu. 2025a. Making text embedders few-shot learners. In *International Conference on Learning Representations*, volume 2025, pages 38107–38124.

Yuan Li, Qi Luo, Xiaonan Li, Bufan Li, Qinyuan Cheng, Bo Wang, Yining Zheng, Yuxin Wang, Zhangyue Yin, and Xipeng Qiu. 2025b. R3-rag: Learning step-by-step reasoning and retrieval for llms via reinforcement learning. *arXiv preprint arXiv:2505.23794*.

Ryan Liu, Jiayi Geng, Addison J Wu, Ilia Sucholutsky, Tania Lombrozo, and Thomas L Griffiths. 2025a. Mind your step (by step): Chain-of-thought can reduce performance on tasks where thinking makes humans worse. In *International Conference on Machine Learning*, pages 38489–38517. PMLR.

Ye Liu, Rui Meng, Shafiq Joty, Silvio Savarese, Caiming Xiong, Yingbo Zhou, and Semih Yavuz. 2024. Codexembed: A generalist embedding model family

for multiligual and multi-task code retrieval. *arXiv preprint arXiv:2411.12644*.

Yuxiang Liu, Tian Wang, Gourab Kundu, Tianyu Cao, Guang Cheng, Zhen Ge, Jianshu Chen, Qingjun Cui, and Trishul Chilimbi. 2025b. Exploring reasoning-infused text embedding with large language models for zero-shot dense retrieval. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 4981–4985.

Meixiu Long, Duolin Sun, Dan Yang, Junjie Wang, Yecheng Luo, Yue Shen, Jian Wang, Hualei Zhou, Chunxiao Guo, Peng Wei, Jiahai Wang, and Jinjie Gu. 2025. Diver: A multi-stage approach for reasoning-intensive information retrieval. *Preprint*, arXiv:2508.07995.

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

Mandar Mitra, Amit Singhal, and Chris Buckley. 1998. Improving automatic query expansion. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 206–214.

Niklas Muennighoff, SU Hongjin, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. 2024. Generative representational instruction tuning. In *The Thirteenth International Conference on Learning Representations*.

Xubo Qin, Jun Bai, Jiaqi Li, Zixia Jia, and Zilong Zheng. 2025. Tongsearch-qr: Reinforced query reasoning for retrieval. *Preprint*, arXiv:2506.11603.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Stephen Robertson, Hugo Zaragoza, and 1 others. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and trends® in information retrieval*, 3(4):333–389.

Rulin Shao, Rui Qiao, Varsha Kishore, Niklas Muennighoff, Xi Victoria Lin, Daniela Rus, Bryan Kian Hsiang Low, Sewon Min, Wen tau Yih, Pang Wei Koh, and Luke Zettlemoyer. 2025. Reasonir: Training retrievers for reasoning tasks. *Preprint*, arXiv:2504.20595.

Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A Smith, Luke Zettlemoyer, and Tao Yu. 2023. One embedder, any task: Instruction-finetuned text embeddings. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1102–1121.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, and 1 others. Stop overthinking: A survey on efficient reasoning for large language models. *Transactions on Machine Learning Research*.

Jianting Tang, Dongshuai Li, Tao Wen, Fuyu Lv, Dan Ou, and Linli Xu. 2025. Large reasoning embedding models: Towards next-generation dense retrieval paradigm. *arXiv preprint arXiv:2510.14321*.

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

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

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

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

Jinsung Yoon, Junhao Zeng, and Sercan Ö Arık. 2025. Square: Unsupervised retrieval adaptation via synthetic data. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 7283–7297.

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