

# HSEARCH-R1: Learning Compact Value Models for LLM-guided Hierarchical Search via RL

Nilesh Gupta<sup>1\*</sup> Nasib Ullah<sup>2</sup> Jinbin Zhang<sup>2</sup> Rohit Babbar<sup>3</sup> Inderjit S. Dhillon<sup>1,4</sup>

<sup>1</sup>The University of Texas at Austin <sup>2</sup>Aalto University <sup>3</sup>University of Bath <sup>4</sup>Google

## Abstract

Reasoning-intensive retrieval often requires evidence whose relevance is not captured by keyword or embedding similarity alone, exposing a core limitation of traditional IR systems. An emerging direction - LLM-guided hierarchical search - addresses this limitation by letting an LLM directly traverse a semantic tree over the corpus. Such methods so far work well only with a frontier LLM in the search loop, which makes inference expensive. We propose HSEARCH-R1, a distillation recipe for learning compact (4B) LLMs that perform efficient LLM-guided hierarchical search. The recipe distills a frontier teacher’s per-step value estimates from its search trajectories into a much smaller LLM (Qwen3.5-4B) via RL, using a composite reward that matches the teacher’s ranking and score distribution at each step; this outcome-level RL distillation substantially outperforms supervised fine-tuning on teacher outputs. We further propose a retrieval-prior-informed variant of the search that injects a cheap retriever’s shortlist as a prior into the tree and adds robustness when the offline index is stale. The base HSEARCH-R1-4B matches or closely tracks frontier LLMs on the seven static BRIGHT-StackExchange subsets at roughly  $50\times$  lower per-query cost. Adding the retrieval prior brings HSEARCH-R1 to 52.5 nDCG@10 on StackExchange and 45.6 across all 12 BRIGHT subsets, within 1 nDCG of the strongest published ensembled baselines while remaining substantially cheaper.

## 1 Introduction

Search systems are increasingly asked to retrieve evidence for questions whose relevance signal is not captured by keyword overlap or embedding similarity alone. In reasoning-intensive retrieval settings such as deep research, scientific discovery, and technical question answering, a relevant

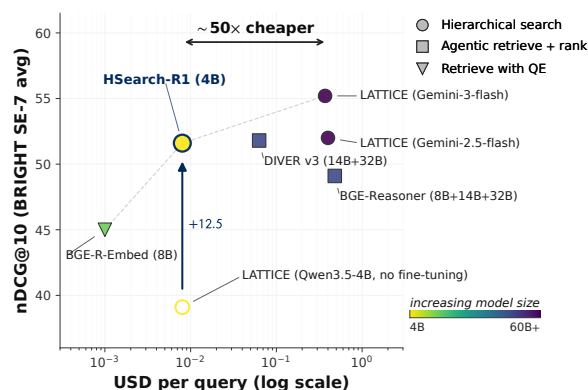


Figure 1: Cost versus quality on BRIGHT-StackExchange (StackExchange-7 avg nDCG@10). HSEARCH-R1 (4B) matches LATTICE-Gemini-2.5-flash at  $\sim 50\times$  lower per-query cost.

document may be useful because it completes an argument, instantiates an abstract condition, or connects to the query through a chain of inference rather than through surface similarity (Su et al., 2025; Shao et al., 2025).

Recent work therefore tries to bring LLM reasoning earlier into the retrieval process. Most dominant line of work keeps the embedding retriever fixed and uses the LLM on the *query side*: the model rewrites the query, generates a hypothetical answer, or runs an agentic retrieve-refine loop before reranking the returned candidates (Gao et al., 2023; Wang et al., 2023; Jin et al., 2025). These methods can improve the query presented to the retriever, but the corpus is still accessed through the retriever’s similarity function, and on BRIGHT most of the benefit of added compute in such pipelines comes from the reranking stage rather than from query expansion (Apparaju and Gupta, 2026). The LLM can suggest where to look, but it does not choose documents directly from the corpus. A second line, LLM-guided search, changes this interface: the corpus is organized into a navigable structure (a tree or a document graph), and a relevance model, a cross-encoder or an LLM,

\*Correspondence: nilesh@cs.utexas.edu

traverses that structure as part of the search algorithm (MacAvaney et al., 2022; Edge et al., 2024; Rathee et al., 2025b; Xu and Chen, 2026; Gupta et al., 2025). This gives the model a chance to inspect the corpus at multiple granularities, semantic regions defined by the structured index and to make relevance decisions that are not forced through a fixed embedding geometry.

In this work we build upon LATTICE (Gupta et al., 2025), a strong instance of the latter paradigm. It constructs a hierarchical index whose leaves are documents and whose internal nodes are LLM-written summaries of the document clusters below them. At query time, a frontier LLM runs best-first search over this tree: it scores the children of the currently expanded node, updates the frontier with the resulting scores, and repeats until the search budget is exhausted. The score assigned to a child is naturally interpreted as a value estimate for that subtree, namely an estimate of how much relevant evidence can be found below that node. Under this view, LATTICE is a learned-value tree search method over a corpus index, where the value function is supplied by a frontier LLM reading node descriptions.

The main obstacle to deploying this form of search is cost. LATTICE calls an LLM at every search step, and a typical query requires 20 to 40 such calls. With a frontier closed-source model in the inner loop, this costs roughly \$0.40 per query, which is one to two orders of magnitude more expensive than common retrieve-then-rerank systems and too costly for high-throughput retrieval (Figure 1). This raises the central question of this paper: can the frontier LLM in LATTICE-style search be replaced by a compact learned value model while preserving the reasoning advantage of hierarchical traversal? Off-the-shelf open models such as Qwen3.5-9B and smaller variants trail frontier LLMs by 10 to 15 nDCG@10 on BRIGHT, a gap large enough to erase the practical value of the lower inference cost.

**Our approach.** We propose HSEARCH-R1, a training recipe for LATTICE-style search: the search algorithm and index are kept, and the contribution is the compact value model and the study of how to train it. We treat the LATTICE per-step score as the object to be learned. Rather than training a compact model to retrieve documents directly, we train a 4B student LLM to imitate the frontier teacher’s value estimates on the slates encountered

during hierarchical search. The distinction is important: the search algorithm consumes a ranked and calibrated score vector over a slate of sibling subtrees, not merely a fluent explanation or a final document ranking. Supervised fine-tuning on the teacher’s JSON output optimizes token-level imitation, but it does not directly optimize the slate-level decision that determines how the frontier evolves. We therefore train the student with RL on teacher rollouts. The reward combines three signals: a slate-nDCG term that rewards matching the teacher’s ranking of children, a slate-KL term that rewards matching the teacher’s score distribution, and a format term that enforces valid structured outputs. This objective supervises the outcome of the student’s generation in the space where search actually uses it.

We also explore a lightweight mechanism for retrieval augmentation. A cheap retriever provides a top- $K$  shortlist, which is converted into a prior over leaves and propagated into the hierarchical search. During traversal, this prior biases the frontier toward subtrees already supported by the retriever; at the end, leaf scores combine the hierarchical-search signal with the retriever signal. This augmentation is not meant to replace LLM-guided search. Instead, it supplies a complementary source of evidence, especially on dynamic-corpus subsets of BRIGHT where the offline tree summaries can become stale and the retriever has access to fresher or more query-specific surface evidence.

**Key results.** On BRIGHT, the base 4B HSEARCH-R1 reaches 51.6/41.1 nDCG@10 on the StackExchange-7 and full 12-subset averages, respectively (Table 1). On the StackExchange subsets, it matches LATTICE-Gemini-2.5-flash (52.0) and is competitive with the strongest ensembled retrieve-then-rerank baselines, DIVER v3 (51.8) and BGE-Reasoner full (52.0), while costing \$0.008 per query instead of \$0.40 for LATTICE-Gemini-2.5-flash. The remaining gap on the full 12-subset average is concentrated on the dynamic-corpus subsets. Adding the retrieval prior raises HSEARCH-R1 to 52.5/45.6, making it the strongest method on the StackExchange average and bringing it within 1 nDCG point of the strongest published ensembled baselines on the full average, while remaining substantially cheaper than those systems (Figure 1, Table 2).

Our controlled comparisons support the underlying search claim. When we fix the LLM and

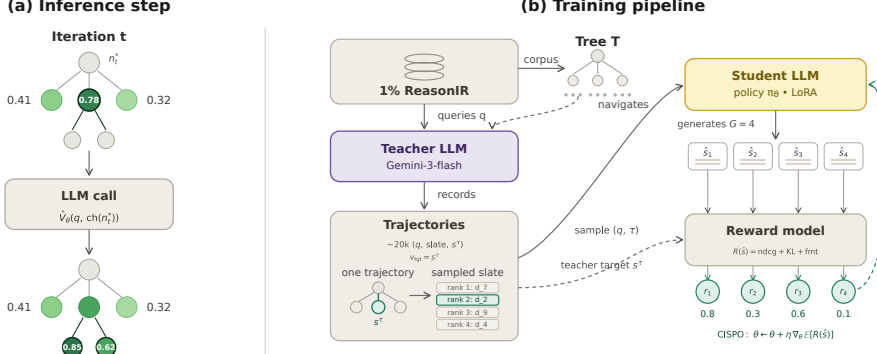


Figure 2: Overview of HSEARCH-R1. **(a) Inference:** at iteration  $t$ , the student LLM scores the children of the top frontier node  $n_t^*$  and the frontier is updated. **(b) Training:** a teacher LLM (Gemini-3-flash) navigates a tree built on 1% of ReasonIR queries, yielding  $\sim 20k (q, \text{slate}, s^T)$  tuples; per slate, the student  $\pi_\theta$  (Qwen3.5-4B, LoRA) generates  $G=4$  score vectors, each scored by  $R(\hat{s}) = R_{\text{ndcg}} + R_{\text{KL}} + R_{\text{fmt}}$  against  $s^T$ .

the retrieval shortlist and vary only the search algorithm, hierarchical search Pareto-dominates flat sliding-window reranking at moderate and high token budgets on every BRIGHT-StackExchange subset (Figure 3). A model-size sweep from 0.8B to 9B shows smooth scaling and identifies 4B as the cost-quality knee: the 9B model adds essentially no further nDCG at higher cost (Table 3). Training ablations show that both slate-level reward terms are necessary. Removing either ranking nDCG or KL to the teacher degrades average nDCG substantially, and pure SFT on teacher outputs lags the RL-trained model by 8 points (Table 4). To our knowledge this is the first systematic study of training a language model to navigate a hierarchical semantic index, in the spirit of the distillation recipes that turned frontier listwise reranking (Sun et al., 2023) into standard open rerankers (Pradeep et al., 2023a,b).

## 2 Related Work

**Hierarchical and tree-based retrieval.** LATTICE (Gupta et al., 2025) is the closest base; it imposes a top-down LLM-built tree and navigates it with a calibrated listwise scorer. RE-TREEVER (Gupta et al., 2026) learns a hierarchy via contrastive routing over embeddings without an LLM in the loop. RAPTOR (Sarathi et al., 2024) recursively summarises clusters bottom-up; GRAPH-RAG (Edge et al., 2024) builds entity / community hierarchies for global summarisation, and HIPPO-RAG (Gutiérrez et al., 2024) a knowledge graph traversed with personalised PageRank. Classical hierarchical ANN structures (HNSW (Malkov and Yashunin, 2020), IVFPQ (Jégou et al., 2011), FAISS (Johnson et al., 2021)) defined what “hi-

erarchical retrieval” meant pre-LLM; we replace the navigation *policy* with a learned LLM-heuristic value function over the tree.

**Value-guided tree search and learned heuristics.** Heuristic tree search guided by a scalar evaluator dates to Pearl (1984) and Korf (1985); the modern neural-heuristic-search line replaces hand-designed heuristics with learned ones (Agostinelli et al., 2019; Yonetani et al., 2021). ALPHAGO (Silver et al., 2016) and ALPHAZERO (Silver et al., 2017) are the most prominent modern instantiations of value-guided tree search but operate in adversarial game environments and pair the value function with Monte-Carlo Tree Search. Our setting (deterministic transitions, fixed corpus tree, no opponent) reduces to greedy best-first heuristic search and inherits the value-distillation abstraction without inheriting MCTS. Within NLP, *tree-of-thoughts* (Yao et al., 2023) and *graph-of-thoughts* (Besta et al., 2024) apply LLM-guided tree search over reasoning states; we apply it over a corpus index, an orthogonal tree structure with explicit relevance labels.

**Distilling frontier rankers and recipe studies.** RANKGPT (Sun et al., 2023) showed that frontier LLMs rerank listwise; RANKVICUNA and RANKZEPHYR (Pradeep et al., 2023a,b) contributed the distillation recipes that compressed this into open 7B models, and RANK1 (Weller et al., 2025) did the same for reasoning-style relevance judgments. Our work plays this role for LLM-guided hierarchical search. Controlled studies of supervision and training choices, such as process versus outcome supervision (Lightman et al., 2024), SFT versus RL (Chu et al., 2025), and the choice

of per-step value target for process reward models (Zhang et al., 2025), are the closest methodological neighbours of our ablations; Zhang et al. (2025) reach an analogous conclusion to our target ablation (§3.2) that rollout-derived Monte-Carlo targets underperform judge-style targets.

**Listwise LLM rerankers.** Listwise LLM scorers (Sun et al., 2023; Pradeep et al., 2023b; Zhuang et al., 2024; Gangi Reddy et al., 2024) rerank flat candidate lists; we apply the same scoring inside a tree search, with anchor-based slate calibration (§3.1).

**Reasoning-intensive retrieval baselines.** On BRIGHT (Su et al., 2025) the strongest single-pipeline systems include DIVER-V3 (Sun et al., 2025), BGE-REASONER-0928, REASONIR-8B (Shao et al., 2025), RADER (Das et al., 2025), REASONRANK (Liu et al., 2026), and RANK1 (Weller et al., 2025). REASONRANK is the closest existing “RL for retrieval ranking” neighbour: like us, it RL-trains a small model with a listwise reward, but it operates on a flat candidate list rather than over a tree.

**Adaptive corpus-graph retrieval.** GAR (MacAvaney et al., 2022), QUAM (Rathee et al., 2025b) and SLIDEGAR (Rathee et al., 2025a) alternate between an initial-pool reranker and graph-neighbour expansion within a fixed LLM-call budget. They are the flat-graph cousin of our hierarchical traversal. RGS (Xu and Chen, 2026) likewise greedily searches an ANN proximity graph guided by a reranker;

**RL training of LLMs for search and value learning.** SEARCH-R1 (Jin et al., 2025), R1-SEARCHER (Song et al., 2025) and RE-SEARCH (Chen et al., 2025) RL-train an LLM to issue search queries and condition on retrieved context. We instead RL-train an LLM to act as a heuristic over a corpus tree. In the value-target literature, MATH-SHEPHERD (Wang et al., 2024), PRM800K (Lightman et al., 2024) and ALPHAMATH (Chen et al., 2024) construct per-step value or reward targets (from human labels, rollouts, or MCTS) and use them for verification or value-guided search; we adapt this idea to slate-level value prediction over a corpus tree. GRPO (Shao et al., 2024) and its CISPO variant give the group-relative baseline that maps naturally onto a slate of children at each tree node.

### 3 Method

This section describes our method in three parts. Section 3.1 reviews the LATTICE search procedure and frames the per-step LLM score as an estimate of a node’s downstream value, which sets up the target the LLM is trying to estimate. Section 3.2 describes how we train a compact LLM to estimate this target accurately, contrasting SFT against an RL recipe. Section 3.3 introduces a simple modification of the search that folds in cheap external retrieval scores, useful when the index is stale (a common problem on dynamic-corpus subsets) or as a low-cost ensembling effect.

**Setting and notation.** A corpus  $\mathcal{D} = \{\ell_1, \dots, \ell_N\}$  is organised into a hierarchical search index  $\mathcal{T}$ , a rooted tree whose leaves correspond to documents in  $\mathcal{D}$  and whose internal nodes carry LLM-generated text descriptions of the cluster of leaf documents under them. We write  $\text{ch}(n)$  for the children of an internal node  $n$ ,  $\text{lv}(n)$  for the leaves under  $n$ , and call the set presented to the LLM scorer at a single step a *slate*; it consists of the children  $\text{ch}(n)$  being expanded plus a small number of additional already-scored sibling nodes that act as calibration anchors so that scores from different slates are placed on a comparable scale. Tree construction is offline and outside the scope of this paper; we use the trees released with LATTICE (Gupta et al., 2025). Given a user query  $q$  and per-leaf relevance labels  $y(\ell, q)$ , the goal at inference is to return a top- $K$  ordered list of leaves that maximises  $\text{nDCG}@K$ , subject to a budget on LLM calls.

**LLM as a listwise scorer.** At each step the LLM receives the query and the slate of cluster descriptions and emits a JSON object with a brief reasoning string and an integer relevance score in  $[0, 100]$  for each candidate (verbatim prompt in Appendix A); listwise scoring with calibration anchors lets scores from different slates be compared on a common scale. We rescale per-child scores  $L(q, \text{ch}(n))$  to  $[0, 1]$  before downstream use.

#### 3.1 LATTICE: hierarchical retrieval guided by an LLM-estimated node value

LATTICE runs a best-first frontier search over  $\mathcal{T}$ . The search maintains a frontier  $\mathcal{F}$  of unexpanded nodes. At each step it pops a node  $n$  from  $\mathcal{F}$ , evaluates an LLM listwise scorer  $L(q, \text{ch}(n))$  on the slate, calibrates the scorer’s outputs to a com-

mon scale across calls (the calibration corrects for prompt-specific bias and lets scores from different slates be compared directly), and inserts each child  $c \in \text{ch}(n)$  back into  $\mathcal{F}$  with a per-node *path-relevance* score

$$\hat{\rho}(c) = \alpha \hat{\rho}(n) + (1 - \alpha) \hat{s}_c, \quad (1)$$

an exponential moving average up the root-to-node path with  $\alpha \in [0, 1]$  (kept at 0.5). Here  $\hat{s}_c$  is the calibrated per-child score in  $[0, 1]$ . After  $T$  iterations the search returns the top- $K$  leaves by  $\hat{\rho}$ .

**Cross-slate comparability.** Raw scores from different calls are not on a common scale, and Eq. 1 does not consume them directly. Each slate contains calibration anchors, already-scored siblings (10 at the leaf level, 2 at internal levels) whose calibrated scores are known; the calibration step regresses the anchors’ new raw scores onto those values and maps the whole slate onto the common scale, absorbing per-slate shift or rescaling. The scorer thus only needs within-slate order and relative spacing right (Table 5).

The LLM’s per-child score  $\hat{s}_c$  can be interpreted as an estimate of the downstream value of expanding the subtree rooted at  $c$ , where value is loosely the relevance of the best leaf, or the top-few leaves, reachable under  $c$ . Concretely we can define an oracle target  $V^*(n; q) = \max_{\ell \in \text{lv}(n)} y(\ell, q)$ ; or any top- $K$ -aware aggregation of the leaf labels under  $n$ . The LLM is trying to approximate  $V^*$  from  $n$ ’s cluster description alone. The calibration step is then a noise model on the raw LLM outputs that maps them to a less biased estimate of  $V^*$ , and the path-relevance EMA in Eq. 1 is a path-aware blend that incorporates evidence from earlier steps along the same trajectory: it mildly discounts a high local score when the parent path was weak, similar in spirit to discounted returns in value estimation.

Search quality strongly depends on how well  $\hat{V}$  approximates  $V^*$  from the node descriptions: frontier LLMs are strong estimators (Gupta et al., 2025), but small open-source LLMs ( $\leq 9\text{B}$ ) trail by 10 or more nDCG points (Table 3), and closing that gap is the goal of this paper.

### 3.2 Training compact LLMs to estimate $V^*$ accurately

**Training data.** We sample  $\sim 1\%$  of ReasonIR’s (Shao et al., 2025) training queries (roughly 1k queries that span the BRIGHT subsets) and run a teacher LLM (Gemini-3-flash) through

the LATTICE search procedure (for 10 iterations with beam size of 2) on a hierarchical index built over the full 100k-document ReasonIR HQ document corpus. For each query we record the teacher’s per-step slate scores and the leaves the teacher’s rollout reaches. This gives us roughly 20 slate-scoring decisions per query, or  $\sim 20\text{k}$  stepwise training examples in total.

**Per-step value targets.** Given a teacher rollout, two natural sources of supervision exist for each child  $c$  scored at any internal step:

1. the teacher’s own per-step score  $s^T(c)$ , and
2. a bottom-up rollout-derived value  $V_\tau(c)$  obtained by aggregating the teacher’s per-leaf scores within the subtree under  $c$ . We use a max aggregation,  $V_\tau(c) = \max_{\ell \in \text{lv}(c)} s^T(\ell) \cdot \mathbb{1}[\ell \text{ visited}]$ , where the indicator restricts the max to leaves the teacher actually scored.

We also experiment with a blended target  $V_{\text{blended}}(c) = \frac{1}{2}(s^T(c) + V_\tau(c))$ . Table 12 compares the three target choices. We find that  $s^T$  and  $V_{\text{blended}}$  are roughly equivalent and both noticeably better than training on  $V_\tau$  alone.

The asymmetry is interesting: even though  $V_\tau$  is the closer proxy for the oracle target  $V^*$ , we hypothesize that the cluster descriptions a student sees at inference are themselves noisy and sometimes under-specific. In that case the value of a node is not perfectly recoverable from its description alone, and asking the student to predict  $V_\tau$  amounts to asking it to predict something outside its observable inputs. By contrast,  $s^T$  is a function of what the cluster description and the query say to the teacher, the same input the student receives at inference, which makes it a more learnable target. We use  $s^T$  as the default target throughout.

**Training method: SFT vs. RL.** Given  $(q, \text{cluster descriptions for } \text{ch}(n), \mathbf{v}^{\text{tgt}})$  tuples, two natural training options are: (a) supervised fine-tuning (SFT) on the teacher’s exact outputs, or (b) RL where the student samples its own outputs and is rewarded against the value targets. We find that SFT does not generalise well; RL with the right reward generalises substantially better (Table 12).

**Supervised fine-tuning.** For SFT we treat the teacher’s full JSON output (which includes a reasoning field and per-child score vector) as

the gold sequence and minimise per-token cross-entropy  $\mathcal{L}_{\text{SFT}} = -\sum_t \log p_\theta(o_t | o_{<t}, x)$ , where  $x$  is the query-and-cluster-descriptions context and  $o = (o_1, \dots, o_T)$  is the teacher’s output sequence.

**RL with composite reward.** For RL we use CISPO (MiniMax, 2025), a clipped-importance-sampling variant of GRPO (Shao et al., 2024) which we found more stable in our setting (cf. (Khattari et al., 2026)). At each step the student samples a group of  $G$  responses for the same prompt; each response is parsed for its predicted score vector  $\hat{s}$ , and rewarded by a composite per-response reward

$$R(\hat{s}) = w_{\text{fnt}} R_{\text{fnt}} + w_{\text{ndcg}} R_{\text{ndcg}}(\hat{s}, \mathbf{v}^{\text{tgt}}) + w_{\text{KL}} R_{\text{KL}}(\hat{s}, \mathbf{v}^{\text{tgt}}), \quad (2)$$

where  $R_{\text{fnt}}$  is a parse-success indicator on the student’s structured output,  $R_{\text{ndcg}}(\hat{s}, \mathbf{v}^{\text{tgt}})$  is the slate-nDCG of the student’s ranking against the value targets (does the student rank children like the teacher?), and  $R_{\text{KL}}(\hat{s}, \mathbf{v}^{\text{tgt}})$  is the negative KL between the student’s and teacher’s softmax score distributions (does the student match the teacher’s calibration?). Precise definitions, floor penalties, and the exact KL form are in Appendix B. We use weights  $(w_{\text{fnt}}, w_{\text{ndcg}}, w_{\text{KL}}) = (0.1, 0.6, 0.3)$ . Both substantive terms are invariant to a constant shift of the scores within a slate: cross-slate scale is handled by the anchor calibration of §3.1, and the reward supervises the two quantities calibration cannot repair. Both terms operate at the slate level, which is the level the downstream search consumes; we believe this slate-level contrastive supervision across the group of generations is what makes the model generalize so much better compared to the SFT model. Table 12 ablates the reward components and shows that removing either the nDCG term or the KL term hurts noticeably.

### 3.3 External retrieval-informed hierarchical search

The value model is only as accurate as the cluster descriptions it sees at inference, and two situations leave headroom for a cheap external retriever to supply complementary signal. (i) **Dynamic-corpus subsets** (BRIGHT LeetCode, AoPS, TheoremQA-Q) apply a different per-query exclusion list to the corpus, so the offline-built summaries become stale: a node summary may still advertise documents that are no longer eligible for the current query. (ii) **Under-informative descriptions** where the LLM

cannot recover the value of a node from the summary alone, but a cheap retriever picks up surface signal the LLM does not see. In both cases we bias the search toward subtrees the retriever already considers promising, while leaving the LLM in charge of fine-grained ordering; we implement this via a soft top- $K$  over retriever scores.

**Soft top- $K$  leaf scoring and frontier gating.** Let  $r(\ell)$  be the score from an off-the-shelf retriever (e.g. BM25) for leaf  $\ell$ , defined on the retriever’s top shortlist and  $-\infty$  otherwise, and let  $s_\ell = \text{minmax}(r(\ell))$ . We use the sigmoid-threshold soft top- $K$  operator of (Gupta et al., 2024), a Lagrangian relaxation of the top- $K$  constraint:

$$v_r(\ell) = \sigma(s_\ell/\tau + t^*), \quad \text{s.t.} \quad \sum_\ell v_r(\ell) = K, \quad (3)$$

where  $\tau > 0$  is a temperature and  $t^*$  is the unique scalar satisfying the constraint (found by binary search since the LHS is monotone in  $t$ ); we use  $K \approx 200$ . We lift to internal nodes by  $v_r(n) = \max_{\ell \in \text{lv}(n)} v_r(\ell)$  and gate the frontier-ordering score multiplicatively:  $\text{score}(n) = \hat{\rho}(n) \cdot v_r(n)$ . Because  $v_r$  is approximately 1 on the top- $K$  retrieval region and approximately 0 outside, this biases the search toward subtrees with a top- $K$  leaf while LLM scores still drive fine-grained ordering.

**Final leaf ranking.** After the search ends, we rank visible leaves by a simple mean of the LAT-TICE path-relevance score and the normalised retriever score:

$$\text{score}_{\text{final}}(\ell) = \frac{1}{2} (\hat{\rho}(\ell) + \text{minmax}(r(\ell))). \quad (4)$$

Hyperparameters  $(K, \tau)$  are tuned once per (retriever, dataset) pair on a held-out ReasonIR training data split. Table 10 ablates the retrieval-augmentation hyperparameters and the choice of retriever.

## 4 Experiments

We aim to address four questions: (1) how does HSEARCH-R1 compare against SOTA systems on BRIGHT (§4.1)? (2) What is the inference cost of HSEARCH-R1 relative to those baselines (§4.1)? (3) Which design choices in the training recipe matter and by how much (§4.2)? (4) How does the recipe scale across student sizes (§4.2)?

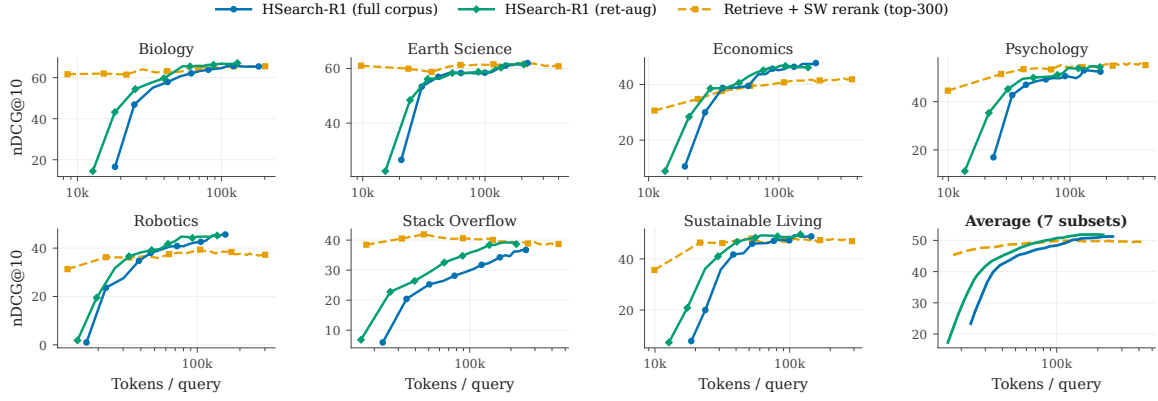


Figure 3: Cost-quality Pareto on 7 BRIGHT-StackExchange subsets; HSEARCH-R1 student LLM held fixed in all curves. *Blue*: hierarchical search, full corpus. *Green*: hierarchical search with BGE-Reasoner top-300 as retrieval prior. *Orange*: sliding-window reranking over the same shortlist.

**Setup.** We evaluate on BRIGHT (Su et al., 2025) (12 reasoning-intensive subsets, partitioned into 7 StackExchange, 2 coding, 3 math/theorem) and 3 traditional IR benchmarks (SciFact, SciDocs, NQ) for breadth (Appendix H). Ablations and model-size scaling are reported on a uniform set of five BRIGHT-StackExchange subsets (Bio, Earth, Eco, Psy, Rob). The student is Qwen3.5-4B with LoRA ( $r=32$ ,  $\alpha=64$ ), trained as in §3.2; the retriever for augmentation is the BGE-Reasoner shortlist (BGE-Reasoner-Embed + BM25, with LLM query rewriting) by default. Trees on BRIGHT subsets are reused from the LATTICE release; the training tree on the 100k ReasonIR HQ corpus is built with Gemini-3-flash and Gemini-3.1-flash-lite following the tree construction algorithm in (Gupta et al., 2025). We use 20 search iterations with beam size 2 unless noted. Full setup and baselines are in Appendix A. Code, the trained checkpoint, and evaluation scripts are available at <https://github.com/nilesh2797/HSearch-R1>.

#### 4.1 Main results

Table 1 reports nDCG@10 on BRIGHT.

**Vs. frontier teachers.** On the 7 static-corpus StackExchange subsets, HSEARCH-R1 (4B) tracks LATTICE-Gemini-2.5-flash within  $\pm 2$  nDCG everywhere (51.6 vs 52.0 SE-7; paired difference not significant, Appendix K) at  $\sim 50\times$  lower per-query cost (Table 2). Against the stronger Gemini-3-flash teacher, the base 4B trails by 3.6 on SE-7 and is within 2 nDCG on 3 of 12 subsets; with retrieval augmentation, this grows to 9 of 12 subsets, with only the longer-context Psy, Rob, Pony still  $>3$  nDCG behind.

**Vs. agentic retrieve-then-rerank pipelines.**

DIVER v3 (14B+32B rerankers, \$0.063/q) and the BGE-Reasoner full pipeline (Qwen3-8B+14B+32B rerankers + LLM embedder + BM25, \$0.71/q) sit  $8\text{--}90\times$  above HSEARCH-R1 on cost. Even without retrieval augmentation, the base 4B matches or outperforms both on 5 of the 12 subsets (Earth, Eco, Psy, Rob, Sus); Augmenting HSEARCH-R1 with the BGE-Reasoner shortlist closes the gap on the dynamic-corpus subsets (LC:  $18.5 \rightarrow 30.1$ , AoPS:  $7.8 \rightarrow 15.3$ , ThT-Q:  $27.7 \rightarrow 45.9$ ) and brings it within 2 nDCG of the ensembled systems on 9 of 12 subsets, with the remaining  $\sim 1$  nDCG 12-avg gap concentrated on Pony (34.0 vs 47.3 / 41.6).

**Cost-quality** Table 2 summarises per-query LLM cost on BRIGHT-StackExchange; per-system token measurements, provider rates, and DIVER/BGE-Reasoner accounting are detailed in Appendices A and F. HSEARCH-R1 (4B) sits at \$0.008/q. The same hierarchical-search algorithm with Gemini-3-flash costs \$0.37/q ( $\sim 46\times$  more) and Gemini-2.5-flash \$0.40/q ( $\sim 50\times$ ), with output dominated by thinking tokens. The strongest retrieve-then-rerank baselines run at intermediate cost on the open-source side: DIVER v3 costs \$0.063/q ( $\sim 8\times$  HSEARCH-R1), spending most of its budget on the 32B groupwise reranker that scores the top-100 shortlist with multiple passes; the ensembled BGE-Reasoner-0928 stack runs at \$0.71/q ( $\sim 90\times$ ), driven by  $\sim 3,500$  reranker calls per query across the 6 retriever  $\times$  size combinations it ensembles. Retrieval augmentation adds only the shortlist cost (\$0.010/q end-to-end), and building the tree is a one-time \$0.16–0.96 per 1k documents that amortizes within  $\sim 1,000$  queries against BGE-Reasoner-0928 (Appendix E).

Table 1: nDCG@10 on BRIGHT. \* = dynamic-corpus subsets. **Bold**: overall best; underline: best among non-Gemini systems. Baselines from their published reports.

| Method                                 | LLM Size   | Bio         | Earth       | Eco         | Psy         | Rob         | Stack       | Sus         | SE Avg      | LC*         | Pony        | AoPS*       | Th-Q*       | Th-T        | Avg         |
|----------------------------------------|------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| <i>Retrieve-then-rerank</i>            |            |             |             |             |             |             |             |             |             |             |             |             |             |             |             |
| Rank-R1-14B                            | 14B        | 31.2        | 38.5        | 21.2        | 26.4        | 22.6        | 18.9        | 27.5        | 26.6        | 9.2         | 20.2        | 9.7         | 11.9        | 9.2         | 20.5        |
| Qwen1.5-7B + InteRank-3B               | 7B + 3B    | 51.2        | 51.4        | 22.4        | 31.9        | 17.3        | 26.6        | 22.4        | 31.9        | 24.5        | 23.1        | 13.5        | 19.3        | 25.5        | 27.4        |
| GPT-4 BM25 + Rank1-32B                 | 32B        | 49.7        | 35.8        | 22.0        | 37.5        | 22.5        | 21.7        | 35.0        | 32.0        | 18.8        | 32.5        | 10.8        | 22.9        | 43.7        | 29.4        |
| ReasonIR + QwenRerank                  | 32B        | 58.2        | 53.2        | 32.0        | 43.6        | 28.8        | 37.6        | 36.0        | 41.3        | 33.2        | 34.8        | 7.9         | 32.6        | 45.0        | 36.9        |
| RaDeR + QwenRerank                     | 32B        | 58.0        | 59.2        | 33.0        | 49.4        | 31.8        | 39.0        | 36.4        | 43.8        | <b>33.5</b> | 33.3        | 10.8        | 34.2        | 51.6        | 39.2        |
| XRR2 (GPT-4 BM25 + gemini-2.5-flash)   | -          | 63.1        | 55.4        | 38.5        | 52.9        | 37.1        | 38.2        | 44.6        | 47.1        | 21.9        | 35.0        | 15.7        | 34.4        | 46.2        | 40.3        |
| ReasonRank                             | 32B        | 62.7        | 55.5        | 36.7        | 54.6        | 35.7        | 38.0        | 44.8        | 46.9        | 29.5        | 25.6        | 14.4        | 42.0        | 50.1        | 40.8        |
| <i>Agentic-retrieve-then-rerank</i>    |            |             |             |             |             |             |             |             |             |             |             |             |             |             |             |
| BGE-Reasoner (no ensemble)             | 32B        | 64.3        | 62.6        | 45.3        | 52.7        | 44.3        | 41.5        | 46.9        | 51.1        | 14.4        | 30.2        | 14.2        | 41.7        | 47.5        | 42.2        |
| BGE-Reasoner (full)                    | 8B+14B+32B | 68.5        | <b>66.4</b> | 40.6        | 53.1        | 43.2        | 44.1        | 47.8        | 52.0        | 29.0        | 41.6        | <b>17.2</b> | <b>46.5</b> | <b>58.4</b> | 46.4        |
| DIVER v3                               | 14B+32B    | 66.0        | 63.7        | 42.4        | <u>55.0</u> | 40.6        | <b>44.7</b> | <b>50.4</b> | 51.8        | 32.5        | <b>47.3</b> | <b>17.2</b> | 46.4        | 55.6        | <b>46.8</b> |
| <i>LLM-guided hierarchical search</i>  |            |             |             |             |             |             |             |             |             |             |             |             |             |             |             |
| LATTICE (Gemini-2.5-flash)             | -          | 66.3        | 63.0        | 47.4        | 54.0        | 47.6        | 37.6        | 48.2        | 52.0        | 19.9        | 34.0        | 12.0        | 38.0        | 47.9        | 43.0        |
| LATTICE (Gemini-3-flash)               | -          | 68.4        | 65.0        | 45.8        | <b>61.0</b> | <b>53.9</b> | 42.2        | <b>50.4</b> | <b>55.2</b> | 22.9        | 43.2        | 13.7        | 44.0        | 46.0        | 46.4        |
| <b>HSEARCH-R1 (4B)</b>                 | 4B         | 65.2        | 63.3        | <b>47.7</b> | 54.8        | <u>45.8</u> | 36.4        | 48.0        | 51.6        | 18.5        | 30.1        | 7.8         | 27.7        | 48.4        | 41.1        |
| <b>HSEARCH-R1 (4B) + retrieval aug</b> | 4B         | <b>69.8</b> | 66.0        | 44.5        | 53.7        | 42.9        | 41.2        | 49.2        | <u>52.5</u> | 30.1        | 34.0        | 15.3        | 45.9        | 54.8        | 45.6        |

Table 2: Per-query LLM cost on BRIGHT-StackExchange. The ret-aug row includes the BGE-Reasoner shortlist cost, dominated by its LLM query rewriter (\$0.0022/q, Appendix G); retrieval itself is negligible. One-time tree-construction cost is in Appendix E. Pricing details in Appendix A.

| Method                           | In/q | Out/q     | Steps/s    | USD/q        |
|----------------------------------|------|-----------|------------|--------------|
| <b>HSEARCH-R1 (4B)</b>           | 156k | 25k       | <b>2.5</b> | <b>0.008</b> |
| <b>HSEARCH-R1 (4B, fast)</b>     | 156k | <b>9k</b> | <b>4.8</b> | <b>0.006</b> |
| <b>HSEARCH-R1 (4B) + ret-aug</b> | 158k | 33k       | <b>2.5</b> | <b>0.010</b> |
| DIVER v3                         | 243k | 44k       | —          | 0.063        |
| BGE-Reasoner-0928                | 3.0M | 882k      | —          | 0.71         |
| LATTICE (G3-flash)               | 163k | 97k       | —          | 0.37         |
| LATTICE (G2.5-flash)             | 164k | 145k      | —          | 0.40         |

**Controlled hierarchical search vs. flat retrieve-then-rerank.** Figure 3 isolates the contribution of the hierarchical search procedure itself by holding the LLM fixed (the same Qwen3.5-4B HSEARCH-R1 student in all curves) and varying the search algorithm. The flat retrieve-then-rerank baseline uses the HSEARCH-R1 student as a sliding-window listwise ranker over the BGE-Reasoner top-300 shortlist; HSEARCH-R1 (ret-aug) runs hierarchical search with the same shortlist as a retrieval prior (without ensembling score with retrieval); HSEARCH-R1 (full corpus) runs hierarchical search over the entire corpus. Setup details (window size, overlap, prompts) are in Appendix J. The plot shows that at low LLM budgets the flat reranker is ahead because it inherits the retrieval-only nDCG for free, but as the budget grows the hierarchical-search curves overtake it on most subsets and at the high-budget end, hierarchical search dominates the flat reranking on most subsets, with

the largest gaps on Economics (+7), and Robotics (+8); the full-corpus variant ends in the same band but takes more iterations to get there.

## 4.2 Ablations and model-size scaling

Ablations are reported on five BRIGHT-StackExchange subsets (Bio, Earth, Eco, Psy, Rob). Retrieval-augmentation and BEIR-transfer ablations are in Appendix H.

**Model-size scaling.** Table 3 sweeps the student backbone from 0.8B to 9B at the same recipe. The recipe lifts every backbone size by 20–35 nDCG@10 on average over its base. Qwen3.5-4B reaches 55.4 average from a 43.6 base; Qwen3.5-9B saturates at 55.5 with no measurable further gain, marking the 4B as the cost-quality knee. The trained 4B and 9B both match Gemini-2.5-flash and trail Gemini-3-flash (teacher) by  $\sim 3$  points.

Table 3: Model-size scaling on BRIGHT-SE (nDCG@10).

| Backbone                                  | Bio         | Earth       | Eco         | Psy         | Rob         | Avg         |
|-------------------------------------------|-------------|-------------|-------------|-------------|-------------|-------------|
| Qwen3-0.8B (base)                         | 2.9         | 2.5         | 0.3         | 1.5         | 0.9         | 1.6         |
| + HSEARCH-R1                              | 44.7        | 42.4        | 25.1        | 32.1        | 28.7        | 34.6        |
| Qwen3-2B (base)                           | 16.0        | 10.0        | 13.5        | 9.6         | 4.3         | 10.7        |
| + HSEARCH-R1                              | 52.2        | 52.1        | 30.3        | 43.6        | 26.9        | 41.0        |
| Qwen3.5-4B (base)                         | 53.3        | 51.2        | 38.9        | 42.7        | 31.8        | 43.6        |
| + HSEARCH-R1                              | <b>65.2</b> | 63.3        | <b>47.7</b> | 54.8        | <b>45.8</b> | 55.4        |
| Qwen3.5-9B (base)                         | 52.4        | 54.7        | 40.6        | 49.8        | 34.5        | 46.4        |
| + HSEARCH-R1                              | 64.7        | <b>64.0</b> | 46.4        | <b>59.4</b> | 43.1        | <b>55.5</b> |
| <i>Frontier LLMs (LATTICE, reference)</i> |             |             |             |             |             |             |
| Gemini-2.5-flash                          | 66.3        | 63.0        | 47.4        | 54.0        | 47.6        | 55.7        |
| Gemini-3-flash                            | 68.4        | 65.0        | 45.8        | 61.0        | 53.9        | 58.8        |

Table 4: Training-recipe ablation (avg nDCG@10 over 5 BRIGHT-SE subsets). Per-subset breakdown in Appendix I.

| Variant                                                  | Avg         |
|----------------------------------------------------------|-------------|
| <b>Default (full recipe)</b>                             | <b>55.4</b> |
| <i>Training method</i> (target = $s^T$ , default output) |             |
| Base (no fine-tuning)                                    | 43.6        |
| SFT (cross-entropy on teacher tokens)                    | 47.4        |
| CISPO, $R_{KL}$ only ( $w_{ndcg}=0$ )                    | <b>51.9</b> |
| CISPO, $R_{ndcg}$ only ( $w_{KL}=0$ )                    | 43.1        |
| <i>Training target</i> (CISPO full, default output)      |             |
| $\mathbf{v}^{tgt} = V_{blended}$                         | <b>55.1</b> |
| $\mathbf{v}^{tgt} = V_\tau$ (rollout-aggregate)          | 52.6        |
| <i>Output format</i> (CISPO full, target = $s^T$ )       |             |
| JSON scores only (no-reasoning, $\sim 2\times$ faster)   | <b>53.0</b> |

**Training recipe.** Table 4 ablates three axes of the recipe relative to the default (55.4 average) shown at the top of the table.

**Training method.** The untrained Qwen3.5-4B base reaches 43.6 average. SFT on the teacher’s full JSON tokens lifts to 47.4, leaving an 8-point gap to the default. CISPO with the  $R_{KL}$  term alone (KL-to-teacher matching) closes most of the gap (51.9), and adding the slate- $R_{ndcg}$  term against teacher scores brings the recipe to its default 55.4. The  $R_{ndcg}$ -only ablation (no KL term) collapses to 43.1, below SFT, confirming that the two reward terms are complementary: KL keeps the score distribution well-calibrated, and ranking-nDCG sharpens the slate order. All of these differences are significant under a paired bootstrap (Appendix K).

**Training target.** The teacher’s per-step scores  $s^T$  (default, 55.4) match the blended target  $V_{blended} = \frac{1}{2}(s^T + V_\tau)$  (55.1) and beat the rollout-aggregate  $V_\tau$  alone (52.6) by  $\sim 3$  points. As discussed in the Section 3.1, we believe this difference is because  $V_\tau$  depends on unobserved inputs, while  $s^T$  is a function of the same inputs the student receives. We default to  $s^T$  for simplicity.

**Calibration anchors.** Table 5 varies the number of already-scored siblings inserted into each slate at inference (§3.1). Removing anchors degrades HSEARCH-R1 monotonically (Bio 65.2 $\rightarrow$ 34.7, Eco 47.7 $\rightarrow$ 25.9 from 10/2 to 0/0), confirming that anchors, not raw scores, carry cross-slate comparability; LATTICE reports the same sensitivity for the frontier teacher (Gupta et al., 2025).

**Output format.** Dropping the JSON reasoning field at the same trained checkpoint halves wall-clock latency at a  $\sim 2$  nDCG@10 cost (53.0 vs

Table 5: Sensitivity of HSEARCH-R1 (4B) to the number of calibration anchors per slate (nDCG@10).  $L/I$  = already-scored leaf/internal siblings inserted into each slate. Removing anchors degrades search monotonically, mirroring the frontier teacher’s sensitivity reported for LATTICE.

| Anchors ( $L/I$ ) | Bio  | Eco  |
|-------------------|------|------|
| 10/2 (default)    | 65.2 | 47.7 |
| 3/2               | 55.0 | 40.7 |
| 0/2               | 39.6 | 30.8 |
| 0/0 (no anchors)  | 34.7 | 25.9 |

55.4); this is the fast operating point shown in Table 2 and offers a useful latency-quality tradeoff for deployment.

## 5 Conclusion

We introduced HSEARCH-R1, a compact value model for LLM-guided hierarchical search that replaces the frontier LLM used in LATTICE-style traversal with a 4B student trained from teacher rollouts. Our results show that slate-level RL is critical for this distillation, substantially outperforming supervised fine-tuning and matching frontier-LLM hierarchical search on BRIGHT’s static StackExchange subsets at roughly  $50\times$  lower per-query cost. We further showed that a cheap retrieval prior improves robustness on dynamic-corpus subsets where offline tree summaries may be stale. Overall, HSEARCH-R1 makes LLM-guided hierarchical search more practical by transferring expensive search-time value estimates into a compact model while preserving most of the retrieval quality.

## Limitations

**Stale indices.** The semantic tree is built once offline, and its cluster summaries become stale on dynamic corpora; the retrieval-augmentation step softens this but does not solve it, and the dynamic-corpus BRIGHT subsets remain the weakest part of our results. Index refresh is cheap in principle (keyword generation, assignment, and summarization are local to the affected root-to-leaf paths; Appendix E), but we do not measure refresh policies or their effect on search quality.

**Dependence on LLM-built trees.** The method inherits the quality of the LLM-written cluster summaries: a node whose description under-represents its leaves cannot be found by any value model, and the student is trained to match the teacher’s reading of those descriptions rather than the underlying documents. Tree construction is a one-time cost of \$0.16–0.96 per 1k documents with the models used here (Appendix E); for much larger corpora we recommend hybrid construction (embedding-based clustering for the lower layers, LLM summarization only for the top), which we do not evaluate.

**Student-rollout distribution shift.** The student is trained only on slates visited by the teacher’s search; at inference its own scoring errors can steer the search to slates that were never in the training distribution, and errors can compound along a trajectory. Our evaluation already runs on trees that were never seen in training (Appendix A), and the beam with a persistent frontier tolerates local misrankings, but we do not study DAgger-style training on student rollouts, which is the natural fix.

**Teacher access and scope.** The recipe assumes a strong teacher with API access; a strong open-source LLM could substitute with degraded trajectories, but we did not study self-distillation. The recipe is tuned for reasoning-intensive retrieval: on classical IR (parts of BEIR) the value-prediction view is less differentiated from flat retrieve-then-rerank and the cost advantage shrinks. A single retriever that handles both regimes within one inference budget is open work.

## References

Forest Agostinelli, Stephen McAleer, Alexander Shmakov, and Pierre Baldi. 2019. Solving the Ru-

bik’s cube with deep reinforcement learning and search. *Nature Machine Intelligence*, 1(8):356–363.

Sreeja Apparaju and Nilesch Gupta. 2026. [Compute allocation for reasoning-intensive retrieval agents](#). *arXiv preprint arXiv:2603.14635*.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefer. 2024. [Graph of thoughts: Solving elaborate problems with large language models](#). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17682–17690.

Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. 2024. [AlphaMath almost zero: Process supervision without process](#). In *Advances in Neural Information Processing Systems (NeurIPS)*.

Mingyang Chen, Linzhuang Sun, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z. Pan, Wen Zhang, Huajun Chen, Fan Yang, Zenan Zhou, and Weipeng Chen. 2025. [ReSearch: Learning to reason with search for LLMs via reinforcement learning](#). In *Advances in Neural Information Processing Systems (NeurIPS)*.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. 2025. [SFT memorizes, RL generalizes: A comparative study of foundation model post-training](#). In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pages 10818–10838. PMLR.

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 (EMNLP)*, pages 19970–19997, Suzhou, China. Association for Computational Linguistics.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitan, Robert Osazuwa Ness, and Jonathan Larson. 2024. [From local to global: A Graph RAG approach to query-focused summarization](#). *arXiv preprint arXiv:2404.16130*.

Revanth Gangi Reddy, JaeHyeok Doo, Yifei Xu, Md Arafat Sultan, Deevya Swain, Avirup Sil, and Heng Ji. 2024. [FIRST: Faster improved listwise reranker with single token decoding](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8642–8652, Miami, Florida, USA. Association for Computational Linguistics.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023. [Precise zero-shot dense retrieval without relevance labels](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1762–1777.

- Nilesh Gupta, Wei-Cheng Chang, Ngot Bui, Cho-Jui Hsieh, and Inderjit S. Dhillon. 2025. [LLM-guided hierarchical search for end-to-end reasoning intensive retrieval](#). *arXiv preprint arXiv:2510.13217*.
- Nilesh Gupta, Fnu Devvrit, Ankit Singh Rawat, Srinadh Bhojanapalli, Prateek Jain, and Inderjit Dhillon. 2024. [Dual-encoders for extreme multi-label classification](#). In *The Twelfth International Conference on Learning Representations (ICLR)*.
- Shubham Gupta, Zichao Li, Tianyi Chen, Cem Subakan, Siva Reddy, Perouz Taslakian, and Valentina Zantedeschi. 2026. [Hierarchical retrieval at scale: Bridging interpretability and efficiency](#). In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*.
- Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. [HippoRAG: Neurobiologically inspired long-term memory for large language models](#). In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. [Unsupervised dense information retrieval with contrastive learning](#). *Transactions on Machine Learning Research*.
- Hervé Jégou, Matthijs Douze, and Cordelia Schmid. 2011. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128.
- Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. [Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning](#). In *Proceedings of the Second Conference on Language Modeling (COLM)*.
- Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. [Billion-scale similarity search with GPUs](#). *IEEE Transactions on Big Data*, 7(3):535–547.
- Devvrit Khatri, Lovish Madaan, Rishabh Tiwari, Rachit Bansal, Sai Surya Duvvuri, Manzil Zaheer, Inderjit S. Dhillon, David Brandfonbrener, and Rishabh Agarwal. 2026. [The art of scaling reinforcement learning compute for LLMs](#). In *The Fourteenth International Conference on Learning Representations (ICLR)*.
- Richard E. Korf. 1985. Depth-first iterative-deepening: An optimal admissible tree search. *Artificial Intelligence*, 27(1):97–109.
- Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. [Let’s verify step by step](#). In *The Twelfth International Conference on Learning Representations (ICLR)*.
- Wenhan Liu, Xinyu Ma, Weiwei Sun, Yutao Zhu, Yuchen Li, Dawei Yin, and Zhicheng Dou. 2026. [ReasonRank: Empowering passage ranking with strong reasoning ability](#). In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 22007–22031, San Diego, California, USA. Association for Computational Linguistics.
- Sean MacAvaney, Nicola Tonellotto, and Craig Macdonald. 2022. [Adaptive re-ranking with a corpus graph](#). In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management (CIKM)*, pages 1491–1500.
- Yu. A. Malkov and D. A. Yashunin. 2020. [Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs](#). *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836.
- MiniMax. 2025. [MiniMax-M1: Scaling test-time compute efficiently with lightning attention](#). *arXiv preprint arXiv:2506.13585*.
- Judea Pearl. 1984. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley, Reading, MA.
- Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. 2023a. [RankVicuna: Zero-shot listwise document reranking with open-source large language models](#). *arXiv preprint arXiv:2309.15088*.
- Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. 2023b. [RankZephyr: Effective and robust zero-shot listwise reranking is a breeze!](#) *arXiv preprint arXiv:2312.02724*.
- Mandeep Rathee, Sean MacAvaney, and Avishek Anand. 2025a. [Guiding retrieval using LLM-based listwise rankers](#). In *Advances in Information Retrieval: 47th European Conference on Information Retrieval (ECIR 2025), Part I*, volume 15572 of *Lecture Notes in Computer Science*, pages 230–246. Springer.
- Mandeep Rathee, Sean MacAvaney, and Avishek Anand. 2025b. [Quam: Adaptive retrieval through query affinity modelling](#). In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining (WSDM)*, pages 954–962.
- Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. 2024. [RAPTOR: Recursive abstractive processing for tree-organized retrieval](#). In *The Twelfth International Conference on Learning Representations (ICLR)*.
- Rulin Shao, Rui Qiao, Varsha Kishore, Niklas Muenighoff, 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](#). In *Proceedings of the Second Conference on Language Modeling (COLM)*.

- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. [DeepSeekMath: Pushing the limits of mathematical reasoning in open language models](#). *arXiv preprint arXiv:2402.03300*.
- David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489.
- David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. 2017. [Mastering chess and Shogi by self-play with a general reinforcement learning algorithm](#). *arXiv preprint arXiv:1712.01815*.
- Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. 2025. [R1-Searcher: Incentivizing the search capability in LLMs via reinforcement learning](#). *arXiv preprint arXiv:2503.05592*.
- Hongjin Su, Howard Yen, Mengzhou Xia, Weijia Shi, Niklas Muennighoff, Han-yu Wang, Haisu Liu, Quan Shi, Zachary S. Siegel, Michael Tang, Ruoxi Sun, Jinsung Yoon, Serkan O. Arik, Danqi Chen, and Tao Yu. 2025. [BRIGHT: A realistic and challenging benchmark for reasoning-intensive retrieval](#). In *The Thirteenth International Conference on Learning Representations (ICLR)*.
- Duolin Sun, Meixiu Long, Dan Yang, Junjie Wang, Yecheng Luo, Yue Shen, Jian Wang, Hualai Zhou, Chunxiao Guo, Peng Wei, Jiahai Wang, and Jinjie Gu. 2025. [DIVER: A multi-stage approach for reasoning-intensive information retrieval](#). *arXiv preprint arXiv:2508.07995*.
- Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. [Is ChatGPT good at search? investigating large language models as re-ranking agents](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 14918–14937.
- Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. [BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models](#). In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*.
- Liang Wang, Nan Yang, and Furu Wei. 2023. [Query2doc: Query expansion with large language models](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9414–9423.
- Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024. [Math-Shepherd: Verify and reinforce LLMs step-by-step without human annotations](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439.
- Orion Weller, Kathryn Ricci, Eugene Yang, Andrew Yates, Dawn Lawrie, and Benjamin Van Durme. 2025. [Rank1: Test-time compute for reranking in information retrieval](#). In *Proceedings of the Second Conference on Language Modeling (COLM)*.
- Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. 2024. [C-Pack: Packed resources for general Chinese embeddings](#). In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 641–649.
- Haike Xu and Tong Chen. 2026. [Beyond sequential reranking: Reranker-guided search improves reasoning intensive retrieval](#). In *The Fourteenth International Conference on Learning Representations (ICLR)*.
- Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. [Tree of thoughts: Deliberate problem solving with large language models](#). In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pages 11809–11822.
- Ryo Yonetani, Tatsunori Tanai, Mohammadamin Barekatain, Mai Nishimura, and Asako Kanezaki. 2021. [Path planning using Neural A\\* search](#). In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pages 12029–12039. PMLR.
- Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. [The lessons of developing process reward models in mathematical reasoning](#). In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 10495–10516, Vienna, Austria. Association for Computational Linguistics.
- Shengyao Zhuang, Honglei Zhuang, Bevan Koopman, and Guido Zuccon. 2024. [A setwise approach for effective and highly efficient zero-shot ranking with large language models](#). In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 38–47.

## A Experimental setup and baselines

**Benchmarks.** BRIGHT (Su et al., 2025) comprises 12 reasoning-intensive subsets partitioned into 7 StackExchange (Bio, Earth, Eco, Psy, Rob, Stack, Sus), 2 coding (LeetCode, Pony), and 3 math/theorem (AoPS, TheoremQA-Q, TheoremQA-T) sets. LeetCode, AoPS, and TheoremQA-Q expose a per-query corpus exclusion list and are marked as dynamic-corpus subsets. Table 6 reports per-subset query and corpus sizes. The 3 traditional IR benchmarks used for breadth are SciFact, SciDocs, and NQ from BEIR (Thakur et al., 2021). Metrics are nDCG@10 (main paper) and Recall@10/100 (Appendix L).

Table 6: BRIGHT subset sizes used in evaluation. \* marks dynamic-corpus subsets where the effective corpus is filtered per-query.

| Subset             | # queries | corpus size |
|--------------------|-----------|-------------|
| Biology            | 103       | 57,359      |
| Earth Science      | 116       | 121,249     |
| Economics          | 103       | 50,220      |
| Psychology         | 101       | 52,835      |
| Robotics           | 101       | 61,961      |
| Stack Overflow     | 117       | 107,081     |
| Sustainable Living | 108       | 60,792      |
| LeetCode*          | 142       | per-query   |
| Pony               | 112       | 7,894       |
| AoPS*              | 111       | per-query   |
| TheoremQA-Q*       | 194       | per-query   |
| TheoremQA-T        | 76        | 23,839      |

**Baselines.** We compare against four families of systems. *Retrieve-then-rerank:* Rank-R1-14B; Qwen1.5-7B + InteRank-3B; GPT-4 BM25 + Rank1-32B (Weller et al., 2025); ReasonIR-8B (Shao et al., 2025) and RaDeR-7B (Das et al., 2025), each with the zero-shot Qwen2.5-32B-Instruct reranker (QwenR-rerank) of the ReasonIR release; XRR2 (GPT-4 BM25 retrieval + Gemini-2.5-flash reranker); ReasonRank-32B (Liu et al., 2026). These chain a sparse/dense retriever with a cross-encoder reranker. *Agentic retrieve-then-rerank:* BGE-Reasoner (no ensemble, a single reranker on top of the BGE-Reasoner-Embed retrieval shortlist); BGE-Reasoner-0928 (full pipeline, ensembles 6 retriever×reranker combos covering 8B, 14B, and 32B rerankers, BGE-Reasoner-Embed, and BM25); DIVER v3 (Sun et al., 2025) (DeepSeek-R1-Distill-Qwen-14B query expansion + Diver-GroupRank-32B groupwise reranker + a Qwen2.5-32B pointwise reranker). *LLM-guided*

*hierarchical search:* LATTICE (Gupta et al., 2025) with Gemini-2.5-flash and Gemini-3-flash. *Embedding-only:* BM25, Contriever (Izacard et al., 2022), BGE-large (Xiao et al., 2024), BGE-Reasoner-Embed. Baseline numbers are taken from the original publications, or from the authors’ public releases and BRIGHT leaderboard entries where those supersede the paper (RaDeR, ReasonRank); the BGE-Reasoner ablations are reported in (Gupta et al., 2025).

**Search hyperparameters.** Unless noted, all hierarchical-search experiments use beam size  $B = 2$ ,  $T = 20$  iterations, path-relevance EMA coefficient  $\alpha = 0.5$  (Eq. 1), and a 5-anchor calibration slate (4 already-scored siblings inserted alongside the children being scored). Retrieval-augmented search uses the BGE-Reasoner shortlist (BGE-Reasoner-Embed combined with BM25 via reciprocal rank fusion, after an LLM query rewrite step also from BGE-Reasoner) with top- $K = 200$  Trees on BRIGHT subsets are reused unchanged from the LATTICE release; the training tree on the ReasonIR HQ 100k corpus is built with Gemini-3-flash and Gemini-3.1-flash-lite following the algorithm of (Gupta et al., 2025).

**Training data curation.** The training corpus is the ReasonIR HQ document set (100k passages); training queries are sampled from the ReasonIR training queries by selecting roughly the top  $\sim 1\%$  ( $\sim 1k$  queries) chosen to cover the topical breadth of the BRIGHT StackExchange subsets. For each query we run the teacher (Gemini-3-flash) through the LATTICE search procedure ( $B = 2$ ,  $T = 10$ ) over the tree above, and record every per-step slate ( $q, \text{ch}(n), s^T$ ) the teacher emits. With  $\sim 20$  slates per query, this yields  $\sim 20k$  slate-scoring training examples. For each child  $c$  in a slate we also store the rollout-aggregate  $V_\tau(c)$  (max teacher score over visited leaves in the subtree under  $c$ ) and the blended target  $V_{\text{blended}}(c) = \frac{1}{2}(s^T(c) + V_\tau(c))$ . We use  $s^T$  as the default target (Table 12 ablates the three options).

**Train/eval overlap.** The training queries are synthetic ReasonIR queries and the training index is a separate tree over the ReasonIR-HQ document pool; no BRIGHT query or relevance label is used in training. We audited the overlap between the 10k-query teacher-run pool from which the training slates were drawn and all BRIGHT evaluation queries and corpora. Queries: 0 exact and

0 near-duplicate matches (word 8-gram containment  $\geq 0.5$ ) against the 1,384 BRIGHT queries. Documents: ReasonIR-HQ was built over documents from the BRIGHT corpora (Shao et al., 2025), so 95,586 of the 95,827 training-tree documents (99.7%) also occur in a BRIGHT corpus, and 364 of the 4,712 BRIGHT gold documents (7.7%) served as the positive of some synthetic query in the pool, which touches 349 of the 1,384 evaluation queries (108 of them in Pony, whose 43 gold documents make any shared document affect many queries). This is the same document exposure carried by retrievers and rerankers trained on ReasonIR data; it involves neither the evaluation queries nor their labels, and the student sees these documents only through the summaries of a different tree, supervised by the teacher’s scores. Training on a document pool disjoint from BRIGHT would remove this exposure; we leave it to future work. Audit scripts are released with the code.

**Training hyperparameters.** Student: Qwen3.5-4B with LoRA ( $r = 32$ ,  $\alpha = 64$ , dropout 0.05, applied to all linear projections). Optimiser: AdamW with cosine learning-rate schedule, peak learning rate  $1 \times 10^{-5}$ , 20 warm-up steps, weight decay 0.01, gradient clipping at 1.0. Batch: per-device batch size 4 slates, gradient accumulation 6,  $G = 4$  generations per slate, sampling temperature 1.0 with top- $p$  0.95. Max prompt length 16,384 tokens, max completion length 2,048 tokens. Mixed-precision bf16 with gradient checkpointing and Liger kernels; vLLM colocated rollout with importance-sampling correction (token-truncate mode). One epoch over the  $\sim 20k$  slate-scoring examples. Composite reward weights  $(w_{\text{fmt}}, w_{\text{ndcg}}, w_{\text{KL}}) = (0.1, 0.6, 0.3)$  with the parse-failure floor at  $-0.25$ , the KL-reward floor at  $-2.0$ , and  $R_{\text{ndcg}}$  measured at  $\text{nDCG}@10$ . RL algorithm: CISPO (MiniMax, 2025) via the GRPO trainer with reference-KL coefficient  $\beta=0$  (the only KL signal is the explicit  $R_{\text{KL}}$  reward against the teacher’s score distribution) and batch reward scaling. Hardware: 4 $\times$ GH200 (96GB HBM each); end-to-end training takes total  $\sim 66$  GH200-hours.

**LLM scoring prompt.** The slate-scoring prompt used in both training data extraction and HSEARCH-R1 inference asks the LLM for a JSON object with a brief reasoning string, a ranking, and a per-child relevance score in  $[0, 100]$ . The verbatim cluster-level template is below; a leaf-level variant with the same structure but stricter scoring

guidance is used at depth-1 expansions.

You are an intelligent search agent navigating a hierarchical semantic tree of topics. Your mission is to predict the most promising candidates to find the answer to the user’s query using the relevance definition below.

```

**Relevance Definition:**
{relevance_definition}

## USER QUERY

{query}

## CANDIDATES

Here are the candidates, each is identified by a unique `node_id` provided at the very start in [] (e.g., [0]).

{child_node_options}

## YOUR EVALUATION TASK
1. First, identify the essential problem in the query.
2. Think step by step to reason about why each candidate is relevant or irrelevant (based on the relevance definition). Provide this analysis in the `reasoning` field.
3. Rank these passages based on their relevance to the query. Provide your ranking in the `ranking` field.
4. Assign a relevance score from 0 to 100 (based on the relevance definition and the ranking). Provide relevances in the `relevance_scores` field.

## OUTPUT FORMAT
You must provide your response as a single, clean JSON object with three keys: `reasoning`, `ranking`, and `relevance_scores`.

* `reasoning`: a string.
* `ranking`: an array of integers (node_id order).
* `relevance_scores`: an array of [node_id, score] pairs, e.g. [[0, 85], [1, 92], [2, 73]].

```

The relevance\_definition placeholder is BRIGHT’s per-subset relevance rubric (e.g., for Biology, “a passage is relevant if it provides specific biological mechanisms, experimental evidence, or theoretical frameworks that directly address the question”). The placeholder child\_node\_options is the slate of cluster summaries (or, at leaves, candidate passages) being scored, each prefixed with its node\_id.

**Cost protocol.** USD/q figures price input and output tokens separately. For Gemini-flash variants we use Vertex AI list pricing (Gemini-2.5-flash: \$0.25/1M input, \$2.50/1M output; Gemini-3-flash: \$0.50/1M, \$3.00/1M). For open-source

models we use DeepInfra’s published per-model rates: Qwen3.5-4B (the HSEARCH-R1 backbone) at \$0.03/1M input and \$0.15/1M output; Qwen3-14B (used as a conservative upper-bound proxy for Qwen3-8B and DeepSeek-R1-Distill-Qwen-14B, neither of which DeepInfra hosts directly) at \$0.12/1M and \$0.24/1M; Qwen3-32B (proxy for Qwen2.5-32B-Instruct and Diver-GroupRank-32B) at the Artificial Analysis published rate of \$0.15/1M and \$0.59/1M. Token counts in Table 2 are measured from  $n \sim 100$  runs of our search code on BRIGHT-StackExchange; throughput is measured on a single GH200. Appendix F details how the DIVER-v3 per-query cost in Table 2 is obtained: token counts are measured from an actual inference run of AQ-MedAI/Diver-GroupRank-32B on the 7 StackExchange subsets, not extrapolated from prompt-template inspection.

**Licenses.** We use BRIGHT and BEIR under their research-use licenses; Qwen3.5-4B under Apache 2.0; the LATTICE codebase and ReasonIR HQ corpus under their authors’ open-source releases; and the Gemini-flash variants via the Google Vertex AI commercial API in accordance with its terms of service. All use is consistent with these terms.

## B Composite reward: precise definitions

Let a slate consist of  $M$  children with student-emitted scores  $\hat{s} \in [0, 100]^M$  (rescaled to  $[0, 1]$  below) and teacher-derived value targets  $\mathbf{v}^{\text{tgt}} \in \mathbb{R}^M$ . The composite reward in Eq. 2 is the weighted sum of three terms:

**Format reward.**  $R_{\text{fmt}}$  is a parse-success indicator on the student’s JSON output. The full output must (i) parse as a single JSON object, (ii) contain a `relevance_scores` field of the correct length, and (iii) be an array of `[node_id, score]` pairs with scores in  $[0, 100]$ . We set  $R_{\text{fmt}} = 1$  when all three conditions hold and apply the floor penalty  $R_{\text{fmt}} = -0.25$  on parse failure. On parse failure the other two reward terms are also set to 0, so the total reward for that generation collapses to  $w_{\text{fmt}} \cdot (-0.25) = -0.025$ .

**Slate-nDCG reward.**

$$R_{\text{ndcg}}(\hat{s}, \mathbf{v}^{\text{tgt}}) = \text{nDCG}_K(\pi_{\hat{s}}; \mathbf{v}^{\text{tgt}}),$$

where  $\pi_{\hat{s}}$  is the permutation of the  $M$  children induced by sorting  $\hat{s}$  in descending order, and  $\text{nDCG}_K$  is the standard rank-discounted gain at

cutoff  $K=10$  (or  $M$  if  $M < 10$ ), normalised by the ideal DCG of  $\mathbf{v}^{\text{tgt}}$ . The reward is in  $[0, 1]$  and is maximised when the student’s slate ordering exactly matches the teacher’s.

**Slate-KL reward.** Let  $\tilde{s} = \hat{s}/100$  and  $\tilde{v} = \mathbf{v}^{\text{tgt}}/100$  be the per-slate rescaled scores. We form softmax distributions  $p_{\theta} = \text{softmax}(\tilde{s})$  and  $p^T = \text{softmax}(\tilde{v})$  over the  $M$  children and define

$$R_{\text{KL}} = 1 + \max\left(-\text{KL}(p^T \parallel p_{\theta}), \kappa_{\text{floor}}\right),$$

with floor  $\kappa_{\text{floor}} = -2.0$ . The construction reaches its maximum of 1 when  $p_{\theta} \equiv p^T$  and is clipped at  $1 + \kappa_{\text{floor}} = -1.0$  on divergent KL; the floor bounds the per-response contribution of pathologically miscalibrated generations.

**Weights.** We use  $(w_{\text{fmt}}, w_{\text{ndcg}}, w_{\text{KL}}) = (0.1, 0.6, 0.3)$ . Under these weights and on a well-formed generation,  $R(\hat{s}) \in [w_{\text{fmt}} + w_{\text{KL}} \cdot (1 + \kappa_{\text{floor}}), w_{\text{fmt}} + w_{\text{ndcg}} + w_{\text{KL}}] = [-0.2, 1.0]$ .

## C Use of AI assistants

We used a general-purpose LLM coding and writing assistant during the development of this work. The assistant contributed in three areas: (i) drafting and editing portions of the paper’s prose - rephrasing, tightening, and proofreading from author-provided outlines and findings; (ii) implementation support for plotting (the figures in this paper) and data-processing scripts; (iii) implementation support for baseline and method code. All scientific claims, experimental results, numbers, citations, and final phrasing were verified by the authors. The final responsibility for the content rests with the authors.

## D Potential risks

The student inherits whatever biases (topical, linguistic, cultural) the teacher’s per-step value judgments encode, and we do not perform a teacher-side audit. Lowering the inference cost of LLM-guided hierarchical search also lowers the barrier for large-scale automated retrieval over reasoning-intensive corpora, which could enable downstream applications (e.g., large-scale automated legal or scientific evidence mining) with privacy or surveillance implications; appropriate use depends on the deployment context. The system is evaluated on benchmarks with well-defined corpora and gold relevance labels (BRIGHT, BEIR); behaviour on noisy

or adversarial corpora, including failure modes that arise when the offline tree summaries no longer accurately reflect the underlying documents, is not characterised here.

## E Offline tree-construction cost

The per-query costs in Table 2 exclude the one-time cost of building the semantic tree. We reconstruct that cost from the released trees and the build configuration: every internal node of a hybrid/bottom-up tree corresponds to one summarization call whose input is its children’s descriptions and whose output is the stored node description; in the top-down construction, keyword generation runs once per batch of 20 documents, cluster creation once per split event (an internal node with internal children), and cluster assignment once per 50 documents per ancestor split, so a leaf at depth  $d$  is assigned  $d-1$  times. Output token volumes are measured from the descriptions stored in the trees; input volumes from the prompt templates plus the children’s descriptions, capped at the 64 KB prompt limit. Tokens are estimated at 4 characters per token, cluster-creation calls are charged 2k thinking tokens each, and a 10% schema-retry overhead is added. Bulk stages (keyword generation, cluster assignment) use Gemini-3.1-flash-lite (\$0.25/\$1.50 per 1M input/output tokens); cluster creation and summarization use Gemini-3-flash (\$0.50/\$3.00).

Table 7 reports the result. Indexing costs \$0.16 to \$0.96 per 1,000 documents, one-time: \$121 for the 7 StackExchange subsets and about \$699 for all 12 BRIGHT subsets, with LeetCode’s 414k-document corpus accounting for more than half. Against the per-query savings over BGE-Reasoner-0928 (\$0.71/q vs. \$0.010/q for HSEARCH-R1 with retrieval augmentation), the full 12-subset index amortizes within roughly 1,000 queries and a single StackExchange tree within 13 to 60 queries. The trees are shared with LATTICE, so HSEARCH-R1 adds no index cost over the frontier-LLM system it replaces; it lowers the recurring per-query term by  $50\times$  and therefore shortens the amortization horizon. Baseline pipelines also carry unreported offline costs (corpus embedding, and for BGE-Reasoner training an embedder plus three reranker fine-tunes at 8B/14B/32B).

**Index refresh.** Keyword generation and cluster assignment are per-document operations and summaries depend only on a node’s children, so in-

Table 7: One-time LLM cost of building the semantic tree index, reconstructed from the released trees and the build configuration (Appendix E). Bulk stages (keyword generation, cluster assignment) run on Gemini-3.1-flash-lite; cluster creation and summarization on Gemini-3-flash.

| Index                       | Docs    | Int. nodes | USD   | \$/1k docs |
|-----------------------------|---------|------------|-------|------------|
| Training tree (ReasonIR-HQ) | 95,855  | 8,965      | 113.6 | 1.19       |
| Biology                     | 57,359  | 2,485      | 9.0   | 0.16       |
| Earth Science               | 121,249 | 8,762      | 21.8  | 0.18       |
| Economics                   | 50,220  | 3,977      | 11.4  | 0.23       |
| Psychology                  | 52,835  | 4,144      | 11.5  | 0.22       |
| Robotics                    | 61,961  | 4,754      | 12.3  | 0.20       |
| Stack Overflow              | 107,081 | 8,736      | 42.7  | 0.40       |
| Sustainable Living          | 60,792  | 4,726      | 12.6  | 0.21       |
| Pony                        | 7,894   | 1,156      | 2.3   | 0.29       |
| TheoremQA-T                 | 23,839  | 1,690      | 6.2   | 0.26       |
| LeetCode                    | 413,910 | 43,462     | 397.8 | 0.96       |
| AoPS + TheoremQA-Q (shared) | 188,211 | 18,741     | 171.9 | 0.91       |
| All 12 BRIGHT subsets       | 1.19M   |            | ≈699  | 0.59       |

serting or removing documents requires re-running these stages only along the affected root-to-leaf paths; refresh cost scales with the number of changed documents rather than the corpus size. Refreshing summaries lazily (only when a node’s document set changes by more than a threshold) bounds the cost further; we leave a measured study of refresh policies to future work.

## F DIVER v3 per-query cost on BRIGHT-StackExchange

DIVER v3 runs 32 LLM calls per query (2 R1-thinking generations from DeepSeek-R1-Distill-Qwen-14B for query expansion, plus 30 groupwise scoring calls from Diver-GroupRank-32B that score the top-100 retriever shortlist in groups of 20 across 6 repeat epochs); the full v3 stack additionally runs a 100-call pointwise reranker on Qwen2.5-32B-Instruct to reach the published 46.8 nDCG@10. We measure input tokens with the actual model tokenizers on real BRIGHT prompts and we measure groupwise reranker output tokens from a real inference run of the released Diver-GroupRank-32B weights on 2 queries from each of the 7 StackExchange subsets (420 total generations); the reranker emits a mean of 1,083 output tokens per call (median 1,060,  $p_{90}$  2,210), roughly  $2\times$  what a prompt-inspection estimate gives because its “brief” reasoning expands per-document. QExpand output is estimated at  $\sim 2,500$  tokens per call from one R1-distill response in the DIVER repository at a  $5\times$  thinking-to-answer ratio. Both stages are

priced at the Artificial Analysis Qwen3-32B rate of \$0.15 input and \$0.59 output per 1M tokens (retrieved 2026-05-26; the two actual DIVER LLMs are not listed by major commodity providers, and Qwen3-32B is the closest publicly priced same-family, reasoning-capable analog). Table 8 reports the resulting per-subset costs: **\$0.039 per query** averaged across the 7 StackExchange subsets for the groupwise-only interpretation of v3, or **\$0.049 per query** for the full pointwise + groupwise stack; the groupwise reranker accounts for  $\sim 85\%$  of the cost and long-document subsets (Stack, Rob) are roughly  $2\times$  more expensive than short-document subsets because each call has to fit 20 documents into its prompt.

## G BGE-Reasoner-0928 per-query cost on BRIGHT-StackExchange

BGE-Reasoner-0928 runs four stages per query: a Qwen2.5-7B rewriter produces 5 query rewrites; two retrievers (BM25 and a Qwen3-8B embedding encoder) each return a top-2000 against the 5 rewrites; the top-100 from each retriever is reranked by three Qwen3 fine-tunes at 8B, 14B, and 32B parameters; a final score-fusion step is pure arithmetic. Each (retriever, reranker-size) pair runs the published two-pass protocol of 100 reranker calls at sample count 1 plus 30 reranker calls at sample count 16, so  $100+480 = 580$  reranker calls per pair and 3,480 reranker calls per query in total; the reranker therefore dominates cost by orders of magnitude. We measure per-call input tokens by reproducing the exact published reranker prompt template (a fixed system instruction plus the per-subset relevance definition, the original BRIGHT query, and one candidate document), applying the Qwen3 chat template in non-thinking mode (the released 0928 setting), and tokenising on 25 queries  $\times$  80 random corpus documents per subset. Output tokens are measured by running each of the three released reranker fine-tunes on 250 BRIGHT-biology calls each with the published sampling parameters ( $T=0.6$ ,  $k=40$ , 4,096-token output cap); the three sizes land within 3% of each other at 254, 250, and 256 mean tokens per call respectively, confirming that the structured grammar of the prompt (three short analyses plus a score tag) bounds output length more tightly than model size does, so we hold the per-size output constants while letting per-subset input length vary. The rewriter contribution

is modelled as  $5 \times (400 + |q_{\text{orig}}|)$  input tokens and 7,930 output tokens (the sum across the 5 rewrites measured directly from the published rewritten-query dataset using the Qwen2.5 vocabulary); embed encoding contributes  $< \$10^{-4}$  per query. Pricing follows the cost protocol in Appendix A: the 8B and 14B fine-tunes at the DeepInfra Qwen3-14B rate (\$0.12 / \$0.24 per 1M input/output), the 32B fine-tune at the Artificial Analysis Qwen3-32B rate (\$0.15 / \$0.59), retrieved 2026-05-26. Table 9 reports the per-subset breakdown: **\$0.713 per query** averaged across the 7 StackExchange subsets. The 32B reranker is the largest single contributor (\$0.33/q on average) because its per-token output rate is  $2.5\times$  the 14B rate, even though it emits essentially the same number of tokens; long-document subsets (Stack, Rob) are roughly  $1.4\times$  more expensive than short-document subsets because each reranker call must fit a longer document into the same template.

## H Additional ablations

**Retrieval-augmentation.** Table 10 ablates the retrieval-augmentation mechanism on a 7-subset breakdown that adds two dynamic-corpus subsets to the standard five. The retriever (BM25 + BGE-Reason-Embed) alone reaches 43.3 average; plain LATTICE with  $K=100$  reaches 48.8; post-hoc score merging lifts that to 50.4 with the largest gains on the dynamic-corpus subsets (LeetCode 22.6 $\rightarrow$ 30.1, ThT-Q 42.5 $\rightarrow$ 45.9), confirming that the retrieval signal compensates for stale tree summaries. Using the retrieval prior only inside the frontier gate, without the final score merge of Eq. 4, did not improve over plain hierarchical search in a 40-query pilot across eight subsets, so the gain is attributable to the score merge; the gate mainly speeds up convergence (Figure 3).

**BEIR transfer.** Table 11 reports nDCG@10 on three BEIR subsets to verify the recipe transfers beyond reasoning-intensive retrieval. The trained 4B reaches 53.8 average across SciFact, SciDocs, and NQ, on par with the LATTICE-Gemini-3-flash teacher (55.0) and ahead of BGE-large-en-v1.5 (50.7) and BGE-reranker (53.4). The recipe was tuned for reasoning-intensive retrieval and we expect a smaller margin on classical IR; the takeaway is that the recipe transfers without modification.

| Subset               | Groupwise reranker |               | QExpand      |              | Groupwise-only | Full v3       |
|----------------------|--------------------|---------------|--------------|--------------|----------------|---------------|
|                      | in/q               | out/q         | in/q         | out/q (est.) | \$/q           | \$/q          |
| Bio                  | 68,066             | 35,022        | 1,680        | 5,000        | 0.0341         | 0.0407        |
| Earth                | 84,425             | 23,797        | 2,105        | 5,000        | 0.0300         | 0.0369        |
| Eco                  | 102,826            | 35,646        | 1,795        | 5,000        | 0.0397         | 0.0471        |
| Psy                  | 70,334             | 30,543        | 2,005        | 5,000        | 0.0318         | 0.0397        |
| Rob                  | 145,046            | 35,768        | 2,807        | 5,000        | 0.0462         | 0.0624        |
| Stack                | 241,308            | 34,146        | 3,705        | 5,000        | 0.0598         | 0.0737        |
| Sus                  | 75,599             | 32,651        | 2,015        | 5,000        | 0.0339         | 0.0412        |
| <b>7-subset mean</b> | <b>112,515</b>     | <b>32,510</b> | <b>2,302</b> | <b>5,000</b> | <b>0.0394</b>  | <b>0.0488</b> |

Table 8: DIVER v3 per-query token counts and USD cost on the 7 BRIGHT StackExchange subsets, priced at Qwen3-32B Artificial Analysis rates (\$0.15/\$0.59 per 1M input/output). Groupwise reranker tokens are measured by running the released 32B reranker on 2 queries per subset (30 calls each); QExpand and pointwise reranker columns are static analyses of the released DIVER prompt templates on real BRIGHT corpus documents.

| Subset               | Reranker tot. |            | Per-size \$/q |              |              |              |
|----------------------|---------------|------------|---------------|--------------|--------------|--------------|
|                      | in/q (M)      | out/q (k)  | 8B            | 14B          | 32B          | \$/q         |
| Bio                  | 2.36          | 882        | 0.165         | 0.164        | 0.293        | 0.625        |
| Earth                | 2.43          | 882        | 0.168         | 0.167        | 0.297        | 0.633        |
| Eco                  | 2.72          | 882        | 0.180         | 0.178        | 0.311        | 0.671        |
| Psy                  | 2.64          | 882        | 0.176         | 0.175        | 0.307        | 0.661        |
| Rob                  | 4.17          | 882        | 0.238         | 0.236        | 0.384        | 0.860        |
| Stack                | 4.40          | 882        | 0.247         | 0.246        | 0.395        | 0.890        |
| Sus                  | 2.57          | 882        | 0.173         | 0.172        | 0.304        | 0.652        |
| <b>7-subset mean</b> | <b>3.04</b>   | <b>882</b> | <b>0.192</b>  | <b>0.191</b> | <b>0.327</b> | <b>0.713</b> |

Table 9: BGE-Reasoner-0928 per-query token counts and USD cost on the 7 BRIGHT StackExchange subsets. “in/q” and “out/q” sum across the  $3 \times 2 = 6$  (reranker-size, retriever) combos with 580 calls each (3,480 reranker calls per query). Per-size \$/q is the cost contribution of each reranker variant (BM25 + Embed). Total \$/q additionally includes the rewriter (\$0.0022/q) and embed encoding (negligible). Reranker output tokens per call are empirically measured by running each of the three released reranker fine-tunes on 250 BRIGHT-biology calls and held constant across subsets (254, 250, 256 for 8B/14B/32B); per-subset input tokens are computed by tokenising the published prompt template on 25 queries  $\times$  80 random corpus documents per subset. Pricing: 8B and 14B at DeepInfra Qwen3-14B (\$0.12/\$0.24 per 1M), 32B at Artificial Analysis Qwen3-32B (\$0.15/\$0.59 per 1M).

## I Full training-recipe ablation

Table 12 reports the per-subset breakdown of the training-recipe ablation summarised in Table 4 of the main paper.

## J Hierarchical search vs. flat reranking: setup details

The controlled comparison in Figure 3 fixes the LLM (Qwen3.5-4B HSEARCH-R1 student) and varies only the search procedure. The flat retrieve-then-rerank baseline uses the BGE-Reasoner

top-300 shortlist (BGE-Reasoner-Embed + BM25, with LLM query rewriting) and applies sliding-window listwise reranking with window size 20 and a 10-document overlap between consecutive windows, the standard configuration used by ReasonRank and the BGE-Reasoner stack. The per-window scoring prompt is the same as the one HSEARCH-R1 uses inside the hierarchical search loop. We sweep the number of windows for the budget axis and report tokens-per-query on the x-axis, including the one-time LLM query-rewriting cost paid before any reranking happens. The HSEARCH-R1 (ret-aug) variant uses the same top-300 shortlist as a retrieval prior in the hierarchical search so the comparison is apples-to-apples on the candidate set; HSEARCH-R1 (full corpus) runs over the entire corpus without retrieval augmentation.

## K Statistical significance

Several comparisons in the main paper are within 1–2 nDCG, and the BRIGHT subsets have only  $\sim 100$  queries each, so single-run differences should be read with their uncertainty. Table 13 reports paired per-query differences with 95% bootstrap confidence intervals on the ablation subsets. The difference to LATTICE-Gemini-2.5-flash is not distinguishable from zero (+1.4, CI  $[-1.1, +3.9]$ ), while the gap to Gemini-3-flash is ( $-4.0$ , CI  $[-6.7, -1.3]$ ). All training-recipe effects in Table 4 are significant except the choice between  $s^T$  and  $V_{\text{blended}}$  as target ( $-0.2$ , CI  $[-2.0, +1.7]$ ), consistent with our reading that the two are equivalent. Repeated runs of the default configuration on the same subset vary by 1–2 nDCG (e.g., 64.1–66.1 on Biology across four runs), which is the noise floor for the per-subset numbers in Table 1.

Table 10: Retrieval-augmentation ablation on 7 BRIGHT subsets (<sup>†</sup> marks dynamic-corpus). Retriever: merged BM25 + BGE-Reasoner-Embed-8B shortlist. HSEARCH-R1 ( $K=k$ ) restricts search to ancestors of the retriever’s top- $k$  leaves; score-merge applies a post-hoc min-max-weighted ensemble. Metric: nDCG@10.

| Variant                                       | Bio          | Earth        | Eco          | Psy          | Rob          | Leet <sup>†</sup> | TQA-Q <sup>†</sup> | Avg          |
|-----------------------------------------------|--------------|--------------|--------------|--------------|--------------|-------------------|--------------------|--------------|
| BM25 + BGE-Reasoner-Embed-8B (retriever only) | 61.67        | 61.01        | 30.63        | 44.58        | 31.30        | 33.15             | 41.06              | 43.34        |
| HSEARCH-R1 (no retrieval aug)                 | 65.20        | 63.30        | <b>47.70</b> | <b>54.80</b> | <b>45.80</b> | 18.50             | 27.70              | 46.00        |
| HSEARCH-R1 ( $K=200$ , no score merge)        | 67.76        | 63.10        | 45.45        | 54.36        | 46.21        | 19.07             | 43.13              | 48.44        |
| HSEARCH-R1 ( $K=200$ ) + score merge          | <b>69.85</b> | <b>66.02</b> | 44.46        | 53.74        | 42.87        | <b>30.11</b>      | <b>45.93</b>       | <b>50.43</b> |

Table 11: nDCG@10 on a three-set subset of BEIR. LATTICE numbers reported in Table 3 of [Gupta et al. \(2025\)](#).

| Method                   | SciFact | SciDocs | NQ   | Avg  |
|--------------------------|---------|---------|------|------|
| BGE-large-en-v1.5        | 74.6    | 22.6    | 55.0 | 50.7 |
| BGE-reranker             | 74.1    | 17.0    | 69.0 | 53.4 |
| LATTICE (Gemini-3-flash) | 75.8    | 22.4    | 66.8 | 55.0 |
| <b>HSEARCH-R1 (4B)</b>   | 74.5    | 24.1    | 62.9 | 53.8 |

## L Search statistics and recall

Table 14 reports, per subset, the index depth and branching, the number of LLM calls per query (at most  $B \times T$  minus the root, i.e., 39 for  $T=20$ ), the mean slate size including anchors, per-query token consumption, and Recall@10/100 for HSEARCH-R1 (4B) without retrieval augmentation. Recall@100 on the StackExchange subsets is 63–83, so most gold documents are surfaced within the search budget even when they are not ranked in the top 10; the dynamic-corpus subsets and Pony have much lower recall, which is where retrieval augmentation helps most.

## M Comparison to BRIGHT leaderboard entries

The public BRIGHT leaderboard is currently topped by closed-source systems that substantially exceed every published baseline we cite in Table 1. We could not reproduce these entries (no public code, model weights, or full system descriptions are available) and we therefore keep the main-paper comparison restricted to systems with published methods and numbers.

Table 12: Training-recipe ablation on BRIGHT-StackExchange (nDCG@10). The default row at the top is our recipe: CISPO with the full composite reward, slate target  $\mathbf{v}^{\text{tgt}}=s^T$ , and a JSON output with both reasoning and scores. Each block below ablates one axis while holding the other two at default.

| Variant                                                              | Bio         | Earth       | Eco         | Psy         | Rob         | Avg         |
|----------------------------------------------------------------------|-------------|-------------|-------------|-------------|-------------|-------------|
| <b>Default (full recipe)</b>                                         | <b>65.2</b> | <b>63.3</b> | <b>47.7</b> | <b>54.8</b> | <b>45.8</b> | <b>55.4</b> |
| <i>Training method</i> (target = $s^T$ , default output format)      |             |             |             |             |             |             |
| Base (no fine-tuning)                                                | 53.3        | 51.2        | 38.9        | 42.7        | 31.8        | 43.6        |
| SFT (cross-entropy on teacher tokens)                                | 55.4        | 57.7        | 44.3        | 44.3        | 35.5        | 47.4        |
| CISPO, $R_{\text{KL}}$ only ( $w_{\text{ndcg}}=0$ )                  | 59.8        | 60.9        | 43.5        | 55.1        | 40.3        | 51.9        |
| CISPO, $R_{\text{ndcg}}$ only ( $w_{\text{KL}}=0$ )                  | 55.1        | 51.9        | 33.3        | 43.3        | 32.0        | 43.1        |
| <i>Training target</i> (CISPO full composite, default output format) |             |             |             |             |             |             |
| $\mathbf{v}^{\text{tgt}} = V_{\text{blended}}$ (bottom-up blend)     | 66.0        | 64.0        | 47.6        | 52.5        | 45.3        | 55.1        |
| $\mathbf{v}^{\text{tgt}} = V_{\tau}$ (rollout-aggregate only)        | 61.0        | 62.5        | 45.2        | 52.3        | 41.9        | 52.6        |
| <i>Output format</i> (CISPO full composite, target = $s^T$ )         |             |             |             |             |             |             |
| JSON scores only (no-reasoning, $\sim 2\times$ faster)               | 66.5        | 61.2        | 45.6        | 49.6        | 42.1        | 53.0        |

Table 13: Paired per-query differences in nDCG@10 between HSEARCH-R1 (4B) and each comparison system on the ablation subsets (Bio, Earth, Eco, Psy, Rob), with 95% paired-bootstrap confidence intervals over queries pooled across subsets (10,000 resamples). Positive = HSEARCH-R1 better.  $^{\dagger}$  = interval excludes zero. Gemini rows use our controlled reruns of LATTICE on the same trees and calibration (Bio, Earth, Eco, Rob only). Published baselines (DIVER, BGE-Reasoner) do not release per-query outputs and cannot be paired.

| HSEARCH-R1 minus                 | Bio  | Earth | Eco   | Psy   | Rob   | Pooled $\Delta$ [95% CI]         | $n$ |
|----------------------------------|------|-------|-------|-------|-------|----------------------------------|-----|
| LATTICE Gemini-2.5-flash         | -0.7 | +3.1  | +2.3  | –     | +0.5  | +1.4 [-1.1, +3.9]                | 423 |
| LATTICE Gemini-3-flash           | -6.5 | -2.2  | -1.1  | –     | -6.4  | -4.0 [-6.7, -1.3] $^{\dagger}$   | 423 |
| Base Qwen3.5-4B (no fine-tuning) | +9.3 | +12.1 | +11.5 | +12.1 | +14.0 | +11.8 [+9.3, +14.4] $^{\dagger}$ | 524 |
| SFT                              | +7.8 | +5.6  | +3.4  | +10.5 | +10.3 | +7.4 [+5.1, +9.8] $^{\dagger}$   | 524 |
| CISPO, $R_{\text{KL}}$ only      | +3.3 | +2.5  | +4.1  | -0.3  | +5.5  | +3.0 [+1.4, +4.7] $^{\dagger}$   | 524 |
| CISPO, $R_{\text{ndcg}}$ only    | +8.0 | +11.5 | +14.3 | +11.5 | +13.9 | +11.8 [+9.1, +14.5] $^{\dagger}$ | 524 |
| target $V_{\text{blended}}$      | -2.9 | -0.7  | +0.0  | +2.3  | +0.5  | -0.2 [-2.0, +1.7]                | 524 |
| target $V_{\tau}$                | +2.1 | +1.0  | +2.4  | +2.5  | +4.0  | +2.3 [+0.2, +4.6] $^{\dagger}$   | 524 |
| JSON scores only (no reasoning)  | -3.4 | +2.1  | +2.0  | +5.2  | +3.7  | +1.9 [+0.2, +3.7] $^{\dagger}$   | 524 |

Table 14: Search statistics and recall of HSEARCH-R1 (4B, no retrieval augmentation) per BRIGHT subset: tree depth and mean branching factor of the index, LLM calls per query, mean slate size (children plus anchors), input/output tokens per query, and Recall@10/100. \* = dynamic-corpus subsets ( $T=30$ ,  $\alpha=0.75$ ); all others  $T=20$ ,  $B=2$ . Biology statistics are from the bottom-up-tree run; per-query outputs for Pony and Th-T were not retained.

| Subset | #q  | Depth | Branch | Calls/q | Slate | In/q | Out/q | R@10 | R@100 |
|--------|-----|-------|--------|---------|-------|------|-------|------|-------|
| Bio    | 103 | 6     | 24.1   | 39      | 16.9  | 138k | 25k   | 68.4 | 82.1  |
| Earth  | 116 | 7     | 14.8   | 39      | 11.3  | 222k | 22k   | 64.0 | 82.8  |
| Eco    | 103 | 6     | 13.6   | 39      | 11.4  | 173k | 23k   | 51.8 | 70.9  |
| Psy    | 101 | 6     | 13.7   | 39      | 12.7  | 183k | 23k   | 58.5 | 76.0  |
| Rob    | 101 | 6     | 14.0   | 39      | 10.7  | 138k | 22k   | 52.7 | 73.4  |
| Stack  | 117 | 6     | 13.3   | 39      | 12.2  | 282k | 23k   | 47.1 | 63.0  |
| Sus    | 108 | 7     | 13.8   | 39      | 11.9  | 127k | 21k   | 54.0 | 72.4  |
| LC*    | 142 | 7     | 10.5   | 59      | 13.3  | 384k | 38k   | 25.1 | 34.9  |
| AoPS*  | 111 | 8     | 11.0   | 59      | 12.5  | 329k | 39k   | 7.6  | 13.8  |
| Th-Q*  | 194 | 8     | 11.0   | 59      | 12.6  | 373k | 38k   | 31.6 | 39.8  |