arXiv:submit/8024678 [cs.AI] 1 Sep 2026

SCALING RERANKED RETRIEVAL

Connor Shorten Weaviate
Daniel J. Williams Weaviate
Daniel M. Jones Weaviate
Augustas Skaburskas Weaviate
John Trengrove Weaviate
Etienne Dilocker Weaviate
Bob van Luijt Weaviate

ABSTRACT

The accuracy of search results is increasingly purchased at inference time rather than baked into a single model. Measured by result quality, most state-of-the-art systems retrieve a candidate pool, rerank it with a cross-encoder, and increasingly re-order the remaining candidates with a listwise reranker. Each addition spends more compute per query, and each raises the same question of where the next unit of compute is best spent: *depth, stages, or width?* We map all three levers on five subsets of BRIGHT, a benchmark designed to test reasoning-intensive information retrieval, using hybrid retrieval that combines Weaviate BM25 with Snowflake Arctic 2.0 embeddings, three commercial cross-encoders: Cohere rerank-v4.0-pro, Voyage rerank-2.5, and ZeroEntropy zerank-2, and three listwise LLM rerankers from OpenAI: GPT-5.4 Mini, GPT-5.6 Luna, and GPT-5.6 Terra. Depth and stages compose into a quality ladder. Deepening retrieval from k=100 to k=2,000 nearly doubles the share of relevant documents available to later stages from 0.476 to 0.871, and each added stage converts more of that availability into a more accurate ranking. Mean nDCG@10 climbs from 0.181 under hybrid retrieval alone, to 0.424 when the strongest cross-encoder reranks the deepest pool, and 0.470 when a listwise pass re-orders the retained top-20 from the cross-encoder. Increasing depth further at the listwise stage extends the ladder less than expected. Retaining 100 rather than 20 candidates from the cross-encoder places substantially more relevant documents within the listwise stage's reach, yet the listwise pass only gains 0.012 nDCG@10 at five times the inference cost. The case for width comes from the models themselves. At each reranking stage, the three models uniquely solve roughly fifty queries their peers miss. However, equal-weight fusion converts almost none of this disagreement, at best marginally exceeding the best single cross-encoder and falling short of the strongest listwise model. Depth and stages are therefore the reliable levers of today's retrieve-then-rerank systems, while width under static fusion is not, marking query-dependent routing as its promising form for future work. Our experimental code is available on GitHub at `https://github.com/weaviate/scaling-reranked-retrieval`.

1 Introduction

Modern search systems increasingly rank the relevance of objects to a query in stages [30, 54, 6]. A first-stage retriever scans the full corpus and returns a candidate pool of size K, a cross-encoder rescores that pool and retains a shortlist of $N \ll K$ candidates. Increasingly, a third stage re-orders the remaining candidates with a listwise LLM that judges the short list jointly and returns the final ranking of M results with $M \leq N$ [47, 26, 35]. The division of labor follows from cost. The first two stages are both pointwise encoders, scoring each document against the query in isolation. However, they differ starkly in mechanism. Whereas the retriever's scores are derived from offline representations and index traversal, the cross-encoder needs online inference over each pair, exchanging speed for accuracy. The third stage introduces a listwise encoder, judging candidates jointly such that documents are weighed against one another, rather than scored apart. This added context is highly effective, but it comes at a cost that scales with the size of the list. Each stage spends more compute per document on fewer documents. Additional comparison frameworks for relevance ranking include pairwise and setwise rerankers, which judge candidates two at a time or in small subsets, respectively [37, 59]. These occupy intermediate points between isolated scoring and full joint comparison. We restrict this work to

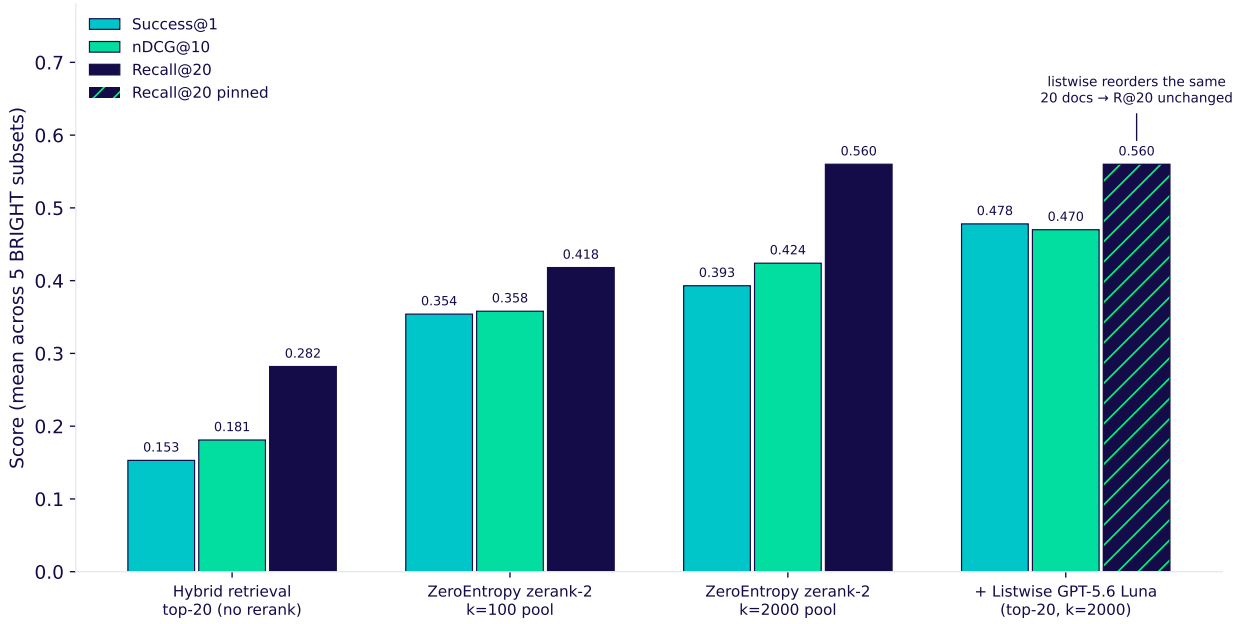


Figure 1: The staged pipeline read as a ladder. Each group adds one lever, a cross-encoder stage, retrieval depth, or a listwise stage, to the group before it. All values are means over the five BRIGHT subsets. The ladder is anchored to ZeroEntropy zerank-2, the cross-encoder carried through the rest of the pipeline. Per-depth best singletons appear in Table 1. The listwise stage reorders the same 20 documents it receives, so its Recall@20 is pinned to the ZeroEntropy zerank-2 input value by construction.

the pointwise-then-listwise pipeline. This pipeline offers three levers for spending additional compute. Depth widens the candidate pools each stage passes forward. Stages spend another pass on what the previous stage kept. Width runs several models at the same stage and fuses their judgments. Where the next unit of compute is best spent: *depth, stages, or width?* is the question this work explores.

Depth is the lever that governs possibility. Concretely, it is the size of each retained list, the retriever's pool K and the cross-encoder's shortlist N. Each stage's retained list is the ceiling the next stage inherits, since a document absent from it cannot be recovered later. A deeper retrieval raises the first ceiling, and a deeper shortlist raises the next. However, depth is not a free good handed to the rest of the pipeline. Each increment of possibility arrives buried in distraction. Nearly every document a deeper pool adds is irrelevant, and the reranker must defend the head of its list against a larger crowd of plausible impostors. *Drowning in Documents* models this cost directly through what its authors term phantom hits, cases in which a reranker draws a spuriously high relevance score on an irrelevant candidate [18]. If each irrelevant candidate independently triggers a phantom hit with some small probability, the expected number of these false positives grows linearly with pool depth, and a large enough pool is therefore expected to place at least one distractor above the gold document. Whether the possibility or the distraction dominates as pools deepen is the first empirical question this work measures.

Stages determine how accurately the ordering converts what depth makes possible. Each stage exists because of something the stage before it cannot do. The retriever cannot read closely, so the cross-encoder reads closely. The cross-encoder cannot compare, so the listwise stage compares. A stage, however, is only an opportunity, and the model performing it may or may not convert that opportunity into a better ordering. Whether these stages are necessary in practice, and how much ordering quality each in fact adds, is the second empirical question this paper measures.

Width is the third lever we explore in our study. At the first stage, width is the norm. Hybrid retrieval fuses BM25 with dense embeddings, motivated in part by the observation that lexical and semantic views of a corpus recover different documents [3]. At the reranking stages, the same reasoning applies and the practice largely stops. Commercial cross-encoders and listwise models are trained by different providers on different data, fail on different queries, and may therefore hold the same kind of complementary evidence that motivates fusion at retrieval.

There is also a statistical motivation for widening the reranking stage. Under the phantom-hit model of [18], suppose an irrelevant candidate is independently assigned a spuriously high relevance score by a reranker with probability λ.

For a pool of K candidates, the expected number of such errors grows as λK. If two rerankers fail independently, however, the probability that the same candidate is mis-scored by both falls to $\lambda_1 \lambda_2$. In this idealized setting, depth increases exposure to distractors linearly while agreement across heterogeneous rerankers suppresses shared errors multiplicatively. The independence assumption and strict-agreement model are deliberately idealized. In practice, a combining rule must grant each model influence where it is uniquely right without paying for that influence everywhere it is wrong, and nothing guarantees that the trade nets out. Whether this multiplicative-suppression intuition carries over under score or rank fusion is the third empirical question this paper measures.

Figure 1 previews the answers we find in our study. The first two levers of depth and stages deliver substantial improvements. Retrieving deeper nearly doubles pool recall, from a mean of 0.476 at $k = 100$ to 0.871 at $k = 2,000$. The added stages convert that expanded pool into a far better ranking, carrying mean nDCG@10 from 0.181 under hybrid retrieval alone to 0.470 under the full pipeline. However, the conversion is uneven. The head of the list stalls early with nDCG@10 and Success@1 both saturating by k=1,000. Deeper pools nonetheless keep improving the retained set, with the strongest cross-encoder's Recall@20 climbing from 0.418 to 0.560 at the deepest pool. Deepening the cross-encoder's output from 20 to 100 results recovers recall to 0.738, illuminating that a substantial share of the missing documents sit just below the initially chosen cutoff. Yet a listwise pass over that deeper top-100 window fails to convert the recovery. The third lever, width, proves far less effective. The models at each reranking stage disagree on roughly fifty queries, but equal-weight fusion converts almost none of that disagreement, finishing at or near the best single model wherever we apply it. Depth and stages are therefore the reliable levers of reranked retrieval, and the disagreement between tested models marks query-dependent routing as a promising opportunity for future work.

Our contributions are as follows:

- We chart a staged pipeline for reasoning-intensive retrieval and measure what each stage buys. Cross-encoding a shallow pool roughly doubles hybrid retrieval's ranking quality, deepening the pool lifts it further, and a listwise pass raises it again, forming a monotonic quality ladder.
- We show that deeper first-stage retrieval nearly doubles candidate availability, while downstream rerankers convert this ceiling asymmetrically. Reranked recall continues to improve with depth, whereas nDCG@10 and Success@1 saturate well before the deepest pools, consistent with the findings from *Drowning in Documents*.
- We show that model choice matters at both reranking stages. Across three commercial cross-encoders over identical pools, ranking quality varies substantially with the model applied, and across three listwise LLM rerankers over an identical cross-encoder shortlist, the two strongest models deliver comparable, substantial gains while the third leaves the mean graded metric unchanged.
- We expand the reranked output cutoff to show that a substantial stratum of gold documents sit between ranks 21 and 100 and then test the natural remedy directly. A listwise pass over the deeper $n = 100$ window buys only marginal mean gains at five times the inference cost, and improves retention on precisely the domains the cross-encoder orders worst while degrading it where the input was already strong. This reproduces the depth and distraction trade-off at the third stage.
- We document complementary errors at both reranking stages. Each group of three models contributes dozens of rank-1 successes that no other model finds, and even the weakest model uniquely solves queries both stronger models miss.
- We show that training-free equal-weight fusion converts almost none of this disagreement. This negative result bounds what static width can deliver and motivates query-dependent routing.

2 Methodology

2.1 Metrics

We utilize three complementary metrics, each matched to a distinct role in a staged pipeline. Our primary metric for ordering quality is nDCG@10, the discounted cumulative gain of the top ten results normalized by the ideal ordering, computed over BRIGHT's binary relevance labels. nDCG@10 rewards placing relevant documents early and continues to credit improvements anywhere in the top ten, and because it aggregates over positions it accommodates the multiple relevant documents that BRIGHT labels per query without the mechanical bounds that afflict single-position metrics. In a staged pipeline, however, ordering is only the terminal concern. Every stage before the last is evaluated by what it retains rather than how it orders, since its output ordering is overwritten by the stage that follows while a document it drops is lost outright. Recall@k, the fraction of a query's relevant documents retained in the top k, is therefore the primary metric for any intermediate stage. Pool Recall@K measures the ceiling first-stage retrieval hands the rerankers, and Recall@20 measures the retained set the cross-encoder hands the listwise stage in most of our experiments. Our

third metric, Success@1, is the fraction of queries whose top result is relevant, the strictest head-of-list criterion. Success@1 and Recall@1 diverge on BRIGHT because queries carry multiple gold documents. With g relevant documents, Recall@1 is bounded by 1/g while Success@1 is not. nDCG@10 sits between the two, graded where Success@1 is strict and position-discounted where Recall@20 is flat, and we treat agreement or disagreement between the three views as informative throughout.

All primary results are reported as means across the five BRIGHT subsets. Subset difficulty varies widely, and the hardest domains to rank, Robotics and Economics, pull each aggregate below the typical subset. The per-subset results in the appendix disclose this spread in full. When we report a gain between two conditions, it is the difference between their cross-subset means. All conditions are evaluated over the same 512 queries, so every reported gain is a paired comparison. To set the scale of the hit-based metrics, each subset contains roughly 100 queries, so a 0.01 difference in Success@1 on a single subset corresponds to one query changing outcome, and a 0.01 difference in a cross-subset mean to roughly one query per subset.

2.2 Evaluation Datasets

We evaluate on five subsets of BRIGHT, a retrieval benchmark built specifically for queries that require reasoning to find relevant results, rather than keyword or semantic matching [46]. BRIGHT's queries are sourced from real StackExchange posts with gold documents sourced from external web pages cited in accepted or highly upvoted answers. We use five subsets drawn from natural-language domains: Biology, Earth Science, Economics, Psychology, and Robotics. The five corpora range from 50,220 documents (Economics) to 121,249 (Earth Science), with Biology at 57,359, Psychology at 52,835, and Robotics at 61,961, and average document lengths between 83.6 words (Biology) and 132.6 (Earth Science). There are 103, 116, 103, 101, and 101 queries in the respective Biology, Earth Science, Economics, Psychology, and Robotics subsets we test with. Of these, 102, 116, 103, 99, and 92 are scored by all three cross-encoders and form the evaluation set of every result in this paper (Section 2.3). They further contain average query lengths of 115, 109, 182, 150, and 819 tokens, respectively. A recent reproducibility study from Sharifymoghaddam et al. [41] integrates BRIGHT baselines into standard IR toolkits and reports a clear capability ladder among first-stage retrievers. Query-side BM25, the benchmark's strongest lexical configuration, and generic neural retrievers, SPLADE-v3 [20] and BGE-large-en-v1.5 [56], perform comparably, all with averages in the range of 0.14 to 0.16 nDCG@10 across BRIGHT. The authors find that large embedding models with 4B parameters post-trained for reasoning-aware relevance more than double these scores. Diver-Retriever-4B [24] reaches 0.291 and Reason-Embed-4B [4] 0.369. At the reranking stage, the study finds that listwise LLM gains shrink as first-stage quality grows, and that a weak reranker can be actively harmful. Applying Qwen3-8B [38] to Reason-Embed-4B's rankings reduces average nDCG@10 by 0.012, with per-subset degradations as sharp as 0.177, whereas stronger rerankers still improve the same input, by $+0.034$ for gpt-oss-120b [31] and $+0.011$ for ReasonRank-32B [23]. Notably, their pipelines omit the intermediate pointwise cross-encoder stage and pass first-stage top-100 directly to the listwise reranker. Their study demonstrates the benefits of embedding and reranking models fine-tuned for reasoning-intensive retrieval. We leave these specialized models outside the scope of this work, focusing instead on general, commercially served retrieval and reranking components.

2.3 Retrieval Systems and Trial Design

Candidates are drawn by Weaviate hybrid search [10], which combines BM25 with Snowflake Arctic 2.0 embeddings [27] under Weaviate's hybrid relative-score fusion, weighting vector search higher than BM25 with $\alpha = 0.75$. The retrieved pool is then reranked by one of three commercial cross-encoders behind their providers' APIs: Cohere rerank-v4.0-pro [7], Voyage rerank-2.5 [52], and ZeroEntropy zerank-2 [33], or by a fusion over several of them. We sweep retrieval depth by retrieving once at k=2,000 and defining each shallower depth as the top-k prefix of that fixed list. Because a cross-encoder scores each (query, document) pair independently of the rest of the pool, every reranker score needed at any depth is already present in the single k=2,000 pass, and the full sweep is derived from one collection with no further API calls. All primary depth comparisons use nested prefixes of a single k=2,000 retrieval. Accordingly, the first-stage availability references reported alongside the reranking results are computed from those same prefixes. This construction holds the initial ranking fixed and isolates the effect of introducing additional candidates as retrieval depth increases. A prefix of the k=2,000 pool is not necessarily identical to a native retrieval issued with `limit`$= k$: the hybrid relative-score fusion normalizes scores over the returned candidate set, so changing the requested limit can change the fused ordering. We therefore treat the nested-prefix experiment as the primary controlled depth sweep and separately rerun native retrieval five times at each limit as a robustness check. Appendix A.2 reports the native-limit results, quantifying both limit-dependent normalization effects and residual run-to-run tie-breaking variation.

We have released the first-stage candidate pools alongside our code as a versioned artifact. For every query we release the full k=2,000 hybrid ranking, keyed to the BRIGHT document identifiers. We further record metadata about the

Weaviate server version (1.38.7) and the collection schema utilized: Snowflake Arctic 2.0 embeddings via text2vec-weaviate, BM25 b=0.75, k1=1.2, word-level tokenization, relative-score fusion at $\alpha = 0.75$. Commercial reranking APIs impose input constraints and exhibit occasional persistent per-query failures that our query set surfaces. Five Robotics queries, the longest in the benchmark at roughly 12,000 to 19,000 characters, exceed Voyage rerank-2.5's per-request limits and cannot be scored by that provider, and one Biology query is likewise unscored by Voyage rerank-2.5. Five Robotics queries, one of them among the five above, consistently returned errors from the ZeroEntropy zerank-2 API across repeated collection attempts, and two Psychology queries consistently returned errors from the Cohere rerank-v4.0-pro API. Twelve distinct queries are affected in total. Every result in this paper, including the singleton depth sweep, the hybrid baselines, and all per-subset appendix tables, is evaluated over the per-subset intersection of queries scored by all three cross-encoders: 102, 116, 103, 99, and 92 queries on Biology, Earth Science, Economics, Psychology, and Robotics respectively, 512 of the 524 benchmark queries, so that every comparison in the paper is paired. The sole exception is the first-stage stability check of Table 11, which re-runs native retrieval as a robustness measurement and is computed over each subset's full query set.

2.4 Listwise Reranking

The listwise experiment isolates ordering within a fixed retained set. We begin with the top-20 returned by ZeroEntropy zerank-2 after reranking the k=2,000 hybrid pool. GPT-5.4 Mini, GPT-5.6 Luna, and GPT-5.6 Terra each receive the query and the same 20 candidate documents, then return a permutation of the corresponding document identifiers. The prompt used for all three models is shown in Appendix E. Additionally, all three models are queried with reasoning effort set to `none`. The listwise rerankers are compared at matched inference settings and observed differences reflect the base model rather than differing reasoning budgets or prompts. Decoding is stochastic, so we run three independent trials per query and align all analyses by trial. A listwise model may promote or demote a document within this set, but it cannot introduce a document that ZeroEntropy zerank-2 placed outside the top-20. Recall@20 is therefore invariant by construction, and the ordering-sensitive metrics, nDCG@10 and Success@1, carry the entire measurement of listwise quality. We report means across trials with the corresponding standard deviations for both.

Beyond the primary top-20 window, we extend the listwise experiment with GPT-5.6 Luna to n=100, over both ZeroEntropy zerank-2's retained top-100 and hybrid retrieval's top-100, under the same prompt, reasoning effort, and trial protocol. These extended-window conditions are evaluated over the same 512 queries as every other condition. A valid response is a permutation of the identifiers 1 through n, each appearing exactly once. At the deeper window, the model sometimes deviates such as repeating an identifier, omitting one, emitting an out-of-range value, or truncating the list early. This illustrates a challenge with structured outputs with LLM decoding [42]. We repair every response with the same two-pass rule. First, scanning the returned list left to right, each in-range identifier is kept at its first occurrence, and repeats and out-of-range values are discarded, leaving a valid partial ranking in the model's own order. Second, any identifiers the model never returned are appended after it, in input order, so that unranked documents retain their upstream order and sit below every document the model did rank. The repaired list is the ranking we evaluate.

2.5 Fusion

We evaluate two training-free blending methods, both applied with equal weights. Reciprocal Rank Fusion (RRF) combines candidates by rank, scoring each document as the sum over input rankings of $1/(\eta + \text{rank})$ with the standard constant $\eta = 60$, agnostic to score scale [8]. Relative Score Fusion (RSF) combines candidates by score, min-max normalizing each reranker's scores to $[0, 1]$ over the candidate pool, recomputed on the restricted pool at every depth, and taking a weighted sum. Where RRF sees only order, RSF preserves the confidence gaps between adjacent documents. At the cross-encoder stage we compare the three singletons against the equal-weight fusion of all three, under both RRF and RSF, with weights identical for every query and domain. At the listwise stage, the models emit permutations rather than scores, so RSF does not apply, and we compare the three singletons against the equal-weight RRF fusion of all three, with fused rankings aligned by trial as in the singleton analyses. We further measure that disagreement between models at a given ranking stage through unique successes. A model receives a unique success at cutoff K on a query when it places a relevant document within its top-K while both other models at the same stage fail. At the cross-encoder stage these counts are computed once per condition. At the listwise stage, where outputs are stochastic, counts are aligned by trial and averaged across the three trials.

3 Experimental Analysis

This section climbs the performance ladder of Figure 1 one rung at a time. Section 3.1 measures how retrieval depth raises the first-stage ceiling. Section 3.2 measures how the three singleton cross-encoders convert pools of growing depth, and Section 3.3 how much a listwise pass over the strongest cross-encoder's retained set improves the head of

the list. Section 3.4 then decomposes the residual gap below the ceiling, documents never retrieved, truncated by the output cutoff, or genuinely misordered, and tests whether a listwise pass over a deeper top-100 window can convert the truncated stratum. The ladder to this point commits each stage to a single model. Section 3.5 widens it, running several models per stage and fusing their judgments.

3.1 Scaling Retrieval

We begin with the first-stage retrieval pool without reranking. Figure 2 shows that hybrid search recall climbs steeply with pool depth on every BRIGHT subset tested. Mean pool Recall@k rises from 0.476 at k=100 to 0.871 at k=2,000. This confirms that retrieval depth is an effective way to increase candidate availability. This result defines the ceiling for every downstream reranker, since a relevant document absent from the top-k pool cannot be recovered later. Even at its deepest setting, the first stage remains highly selective. A pool of 2,000 candidates spans between 1.6% of the corpus on Earth Science and 4.0% on Economics. Deep retrieval compresses the collection by a factor of 25 to 60 while capturing roughly 87 percent of the judged relevant documents. That capture is uneven across domains. Biology and Psychology approach ceilings above 0.9 at k=2,000, while Earth Science, the hardest domain to retrieve, reaches only 0.830, with Robotics close behind at 0.829.

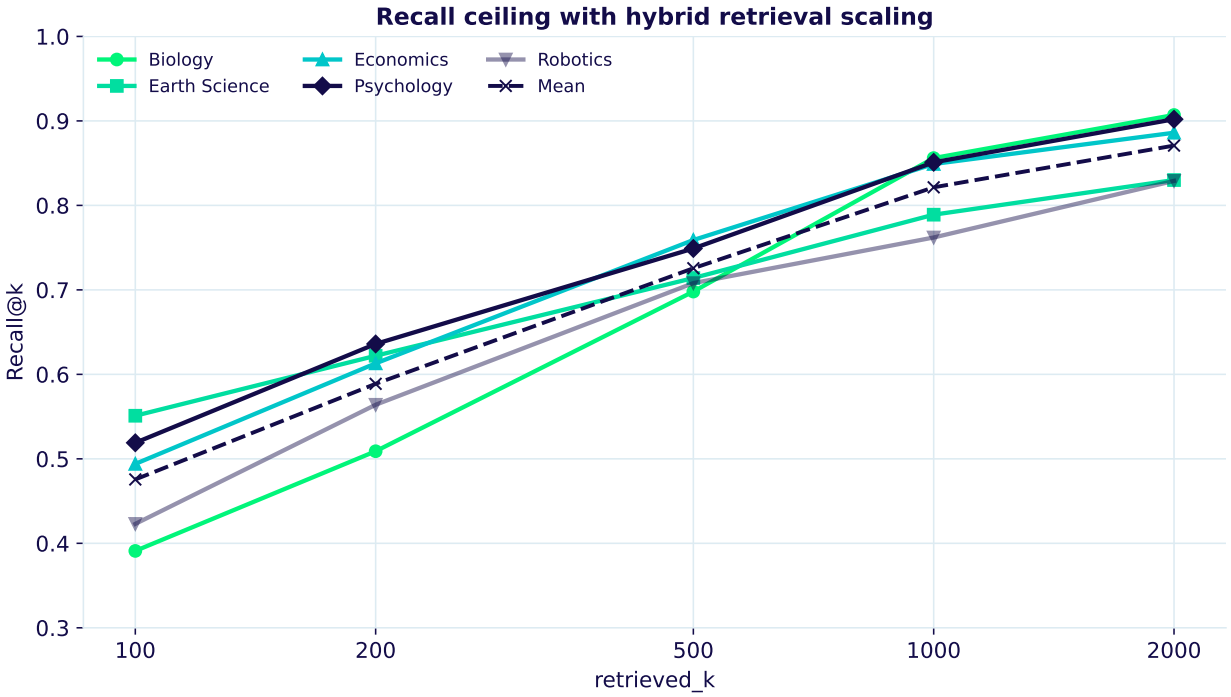


Figure 2: The recall ceiling with hybrid search scaling on BRIGHT subsets.

We verify that this first-stage measurement is stable enough to support the rest of the analysis. Re-running fresh hybrid retrieval five times per subset returns identical Success@1 and Recall@20 on four of the five domains at every depth, and identical Success@1 on all five. Variation appears only in Earth Science's Recall@20, where the largest across-trial standard deviation is 0.001, corresponding to less than half a single-gold query flipping at the cutoff. This residual stochasticity is downstream of relative score fusion, which can produce tied normalized scores whose order depends on the process hash seed, and it bounds the noise floor.

3.2 Singleton Cross-Encoders

How much does a single cross-encoder improve the retrieved pool? Table 1 reports the result across the depth sweep. Over the shallowest pool, k=100, the best cross-encoder roughly doubles the quality of the hybrid ranking, raising mean nDCG@10 from 0.181 to 0.358. Deepening the pool helps further, but where it helps depends on list position. The strongest cross-encoder's mean nDCG@10 climbs from 0.358 at k=100 to 0.423 at k=1,000 and is flat thereafter, ending at 0.424 at k=2,000. Its Success@1 peaks at k=1,000, reaching 0.397 before settling to 0.393. Recall@20 behaves differently, improving monotonically to 0.560 at the deepest pool.

	k=100	k=200	k=500	k=1,000	k=2,000
Hybrid Retrieval Ceiling	0.476	0.589	0.726	0.821	0.871
Success@1					
Cohere rerank-v4.0-pro	0.323	0.324	0.297	0.272	0.247
Voyage rerank-2.5	0.347	0.367	0.376	0.374	0.374
ZeroEntropy zerank-2	**0.354**	**0.383**	**0.393**	**0.397**	**0.393**
nDCG@10					
Cohere rerank-v4.0-pro	0.330	0.351	0.346	0.335	0.313
Voyage rerank-2.5	0.332	0.364	0.379	0.382	0.379
ZeroEntropy zerank-2	**0.358**	**0.396**	**0.415**	**0.423**	**0.424**
Recall@20					
Cohere rerank-v4.0-pro	0.403	0.461	0.500	0.509	0.498
Voyage rerank-2.5	0.414	0.466	0.498	0.512	0.508
ZeroEntropy zerank-2	**0.418**	**0.480**	**0.528**	**0.555**	**0.560**

Table 1: Mean Success@1 (top panel), nDCG@10 (middle panel), and Recall@20 (bottom panel) across subsets achieved by singleton rerankers. The shaded row is the hybrid retrieval ceiling, the first-stage pool recall at each depth, which bounds the Recall@20 any reranker can reach over that pool and is shown alongside the two ordering metrics as a candidate-availability reference. The best performing reranker per depth and metric is highlighted in bold.

Whether depth helps at all, however, depends on the model. ZeroEntropy zerank-2 produces the saturating shape shown in Figure 3, leading the cross-subset mean nDCG@10 at every depth and in twenty-four of the twenty-five (subset, depth) cells, with the single exception at k=100, where Cohere rerank-v4.0-pro edges it on Biology at 0.315 against 0.314. More detailed results can be found in Appendix B.3. Voyage rerank-2.5 follows the same shape at a lower level, saturating at k=1,000. Cohere rerank-v4.0-pro inverts it. Its mean nDCG@10 peaks at 0.351 by k=200 and then falls to 0.313 at k=2,000, below its own shallowest-pool value, with Success@1 falling in parallel from 0.324 to 0.247. This is the depth-dependent regression that *Drowning in Documents* predicts. Deeper pools supply this model with high-scoring distractors faster than they supply recoverable gold documents. The per-subset results, reported in Appendix Tables 12 to 16, sharpen the effect. Cohere rerank-v4.0-pro's collapse is concentrated on Economics, where Success@1 falls from 0.233 to 0.097, Psychology, from 0.394 to 0.212, and Robotics, from 0.228 to 0.141, while on Biology and Earth Science the same model over the same depth sweep is stable or improving. Susceptibility to *Drowning in Documents* is therefore a property of the (model, domain) pair rather than the model alone, and the vulnerable domains are precisely the three that every reranker finds hardest to order.

Success@1 and Recall@20 refine this picture without overturning it. Voyage rerank-2.5 leads at rank 1 on some domains at shallow depth, and Cohere rerank-v4.0-pro remains competitive at Recall@20 even where its head-of-list accuracy collapses. Placing the single best document first and keeping gold documents alive within the top twenty are therefore partially independent capabilities. These aggregate disagreements foreshadow the query-level fact that Section 3.5 quantifies: each model solves queries the others miss. Because ZeroEntropy zerank-2 leads the primary ordering metric at every depth and retains the most at the deepest pool, its top-20 reranked from $k = 2{,}000$ is the input to the listwise experiment of Section 3.3.

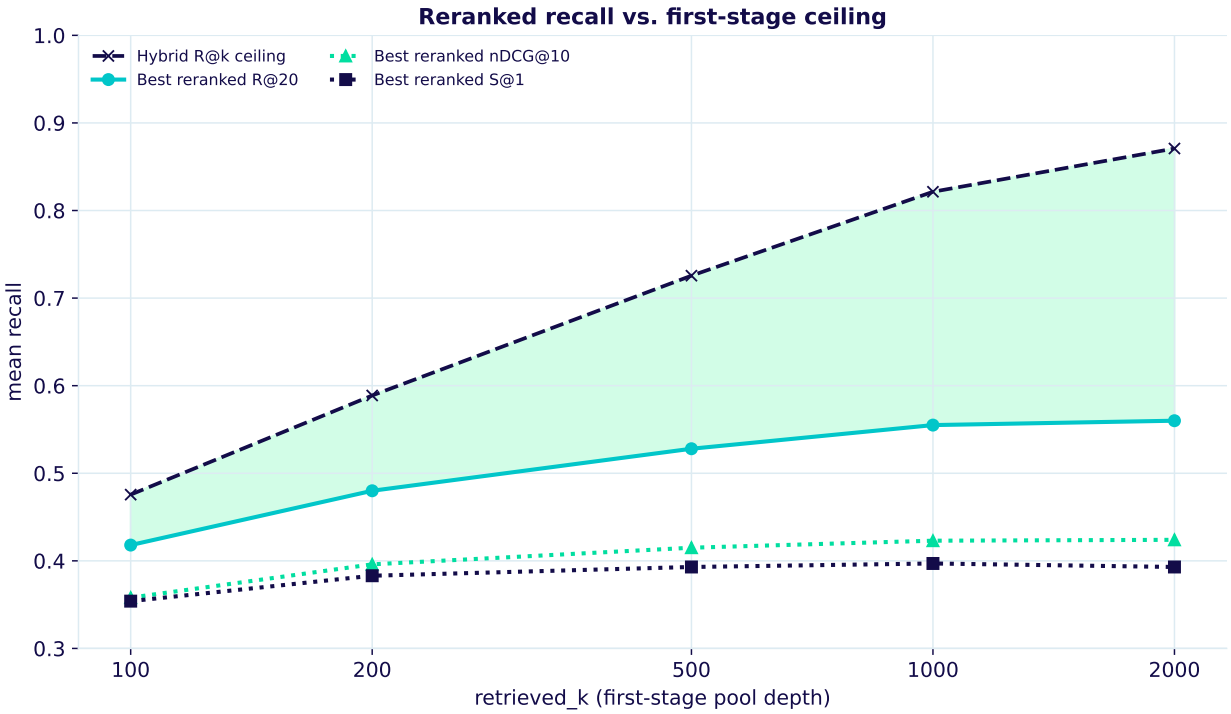


Figure 3: Average reranked performance at output cutoffs of 1 and 20 versus the hybrid first-stage recall ceiling, as first-stage depth grows from k=100 to k=2,000. The retained set, Recall@20, continues to convert deeper pools while the head of the list, Success@1, saturates. Both nevertheless fall increasingly far below the first-stage ceiling as depth grows, and Section 3.4 decomposes this widening gap.

3.3 Singleton Listwise Rerankers

With depth exhausted and cross-encoder ordering saturated, the listwise stage buys another head-of-list gain. Re-ordering ZeroEntropy zerank-2's retained top-20 under the protocol of Section 2.4, GPT-5.6 Luna raises mean nDCG@10 from 0.424 to 0.470 and Success@1 from 0.393 to 0.478, gains of 0.046 and 0.085 respectively. Per-subset results can be found in Appendix C. The rank-1 gain of 0.085 exceeds what any increment of retrieval depth bought anywhere in the sweep, and it comes from a single pass over twenty documents per query. However, the stage is only as good as the model performing it. Figure 4 illustrates the performance of the three listwise rerankers tested. GPT-5.6 Luna and GPT-5.6 Terra lift nDCG@10 by 0.046 and 0.064, while GPT-5.4 Mini nets zero on the graded metric, matching the input's mean of 0.424 while moving individual subsets in both directions. This is consistent with previous works that find weaker listwise rerankers fail to improve strong input rankings [41]. GPT-5.4 Mini's failures can even hide behind the hit metric. On Economics, it gains $+0.049$ Success@1 while nDCG@10 moves by -0.022 on the same queries.

Between the two strong models, GPT-5.6 Terra is the stronger ranker. It leads the graded metric on four of five subsets, with its largest margin on Economics, 0.405 against GPT-5.6 Luna's 0.362, and its average advantage over Luna is 0.018 nDCG@10 and 0.024 Success@1. GPT-5.6 Luna wins Biology and Psychology at rank 1 and costs about a tenth as much per pass, as Section 4.1 details. The ladder of Figure 1 reports GPT-5.6 Luna as its fixed choice on this balance of cost and quality.

3.4 Reranked Depth and the Capture Ceiling

The listwise gains of Section 3.3 raise an immediate question: *would a deeper listwise stage recover more?* The answer depends on where the missed relevant documents sit. If the gold documents absent from the final top-20 were never retrieved, or were scored into the depths of the pool, no realistic window reaches them. If instead they sit just below the current cutoff, a deeper window places them within reach of the listwise reranking stage. To answer this question, we first re-derive every condition with a reranked output cap of 50 and 100 from the same score cache. Shown in Table 2, at a cap of 20 and k=2,000, the best singleton's Recall@20 reaches only 53 to 74 percent of the first-stage ceiling, from 53 percent on Economics to 74 percent on Earth Science. Extending the cap to 100 shows the reading is incomplete.

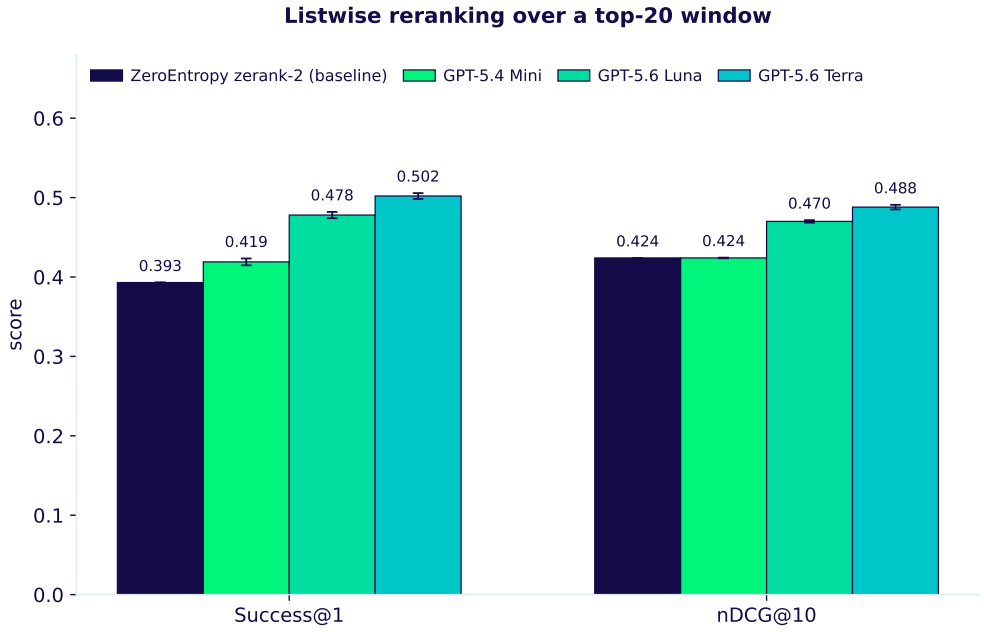


Figure 4: Mean Success@1 and nDCG@10 across all five subsets after single-pass listwise reranking of the ZeroEntropy zerank-2 top-20 at first-stage depth k=2,000. Each model receives the same 20 candidates and may change only their order. Recall@20 is therefore fixed while both ordering metrics move. Error bars show the standard deviation across three trials. Per-subset values appear in Appendix C.

Reranker	**Recall@20**	**Recall@50**	**Recall@100**
Hybrid Retrieval Ceiling	0.871	0.871	0.871
Cohere rerank-v4.0-pro	0.498	0.601	0.682
Voyage rerank-2.5	0.508	0.621	0.688
ZeroEntropy zerank-2	**0.560**	**0.673**	**0.738**

Table 2: Mean recall at output cutoffs of 20, 50, and 100 at first-stage depth k=2,000, against the first-stage recall ceiling. Loosening the output cap recovers part but not all of the gap. Per-subset values appear in Table 27.

ZeroEntropy zerank-2's mean recall rises from 0.560 at a cap of 20 to 0.738 at a cap of 100. The recovered recall diminishes as the cap deepens. Shown in Table 27, ZeroEntropy zerank-2 gains 0.113 between caps of 20 and 50 and 0.065 between 50 and 100. Loosening the cap does not completely close the gap. A mean residual of 0.133 remains below the hybrid retrieval ceiling even at a cap of 100.

This recall gain of 0.178 between output caps of 20 and 100 is opportunity the top-20 listwise experiment never sees, and the mechanism that lifted nDCG@10 by 0.046 on twenty documents seems well placed to convert it. We test this directly with GPT-5.6 Luna re-ordering ZeroEntropy zerank-2's retained top-100 under the protocol of Section 2.4 with n=100. These results are again averaged across three trials. At this window length, GPT-5.6 Luna frequently fails to return an exact permutation. Roughly two thirds of responses deviate, most commonly by repeating a single identifier or truncating the list within the last few positions. The defects are overwhelmingly confined to the tail. Every response across all trials contained at least 20 valid unique identifiers before repair, the median first repair-affected position was 100, and the mechanical repair of Section 2.4 altered a position within the scored top-20 in only 17 responses across both 100-document window conditions and 3 trials. The failure is one of counting to exactly 100, not of ranking, and the results below are insensitive to it. Table 3 reports them.

The deeper window does not scale the listwise gain. At roughly five times the inference cost, Luna over the top-100 reaches a mean nDCG@10 of 0.482 against the top-20 pass's 0.470, a gain of only 0.012, and improves Recall@20 only marginally against the input ordering, from 0.560 to 0.565, gaining on three subsets and degrading on two. Shown in Appendix D, the modest aggregates conceal a coherent per-subset pattern. Relative to the top-20 listwise pass, the

Condition	nDCG@10	Success@1	Recall@20	Recall@100
ZeroEntropy zerank-2 input ordering	0.424	0.393	0.560	**0.738**
Luna over top-20	0.470 ± 0.002	0.478 ± 0.004	0.560	**0.738**
Luna over top-100	**0.482** ± 0.003	**0.486** ± 0.008	**0.565** ± 0.004	**0.738**

Table 3: Deepening the listwise window. GPT-5.6 Luna re-orders ZeroEntropy zerank-2's retained set at k=2,000 under windows of 20 and 100 documents. Recall@100 is constant across conditions because every output is a permutation of the same retained 100 documents, and it is the availability ceiling each condition inherits. Recall@20 is additionally pinned to the input value for the top-20 window. Luna rows are means across three trials $\pm$ across-trial standard deviation. Bold marks the best value per metric. Per-subset results appear in Table 30.

deeper window improves nDCG@10 on every subset, with the largest gains on the domains the cross-encoder orders worst. The trade-off surfaces in retention instead: Recall@20 improves on the three hard-to-order domains and declines on the two easier ones, most sharply on Earth Science at -0.043, where ZeroEntropy zerank-2's top-20 was already strongest. The window helps precisely where the cross-encoder left work undone, consistent with the findings from [41] that listwise gains shrink as input quality grows. Success@1 rises to a mean of 0.486, improving on every subset relative to the input ordering and on three of five relative to the top-20 pass. The deeper window helps the model find the single best document even as it leaves the retained set scarcely better on average, the placement-versus-retention split of Section 3.2 reappearing at the listwise stage. The top-20 window therefore remains the efficient operating point, buying most of the listwise gain at a fifth of the cost.

3.5 Exploring Width in Reranking

The ladder so far commits each stage to a single model, and a single model's errors come with it. Aggregate dominance does not mean per-query dominance. At both reranking stages, the models that lose in aggregate still solve queries the leader misses. Described further in Section 5.3, combining the evidence of multiple rankers is the classical response, with a long history in metasearch and data fusion. Hybrid retrieval already applies it routinely at the first stage, including in our own pipeline. We refer to the same move at a reranking stage as widening it, running several heterogeneous models over the same candidates and combining what they return. What is not established is how much width is worth for modern commercial rerankers, whether for the pointwise cross-encoders or the listwise LLMs that follow them. Section 3.5.1 tests the precondition, whether the models' strengths fall on disjoint queries. Section 3.5.2 measures how much of that disagreement the simplest combining rule, a globally fixed equal-weight blend, converts, leaving per-query selection and adaptive designs [14] to future work.

3.5.1 Complementary Errors at Both Reranking Stages

The models' shifting strengths admit two readings. They may succeed on largely the same queries with different margins, in which case width adds redundant computation, or on different queries, in which case each model holds evidence the others cannot reach. We present a unique success analysis to distinguish the two. A query where one reranker places the gold document within the cutoff while both others miss is recoverable through that model alone. Table 4 reports these counts for the three cross-encoders at k=200 rather than at full depth, because pool depth is not neutral across models. Cohere rerank-v4.0-pro loses roughly a quarter of its Success@1 between k=200 and k=2,000, so counting at full depth would credit Voyage rerank-2.5 and ZeroEntropy zerank-2 with unique recoveries on queries Cohere rerank-v4.0-pro resolves correctly over a healthier pool, misattributing depth sensitivity as complementarity.

The counts support the second reading. The three cross-encoders contribute 52 unique rank-1 successes across 512 queries, and every model contributes on nearly every subset. The structure beneath the total is not uniform. Voyage rerank-2.5's unique contribution collapses from Success@1 to Recall@20 in every subset, falling to at most one, while the Recall@20 leader is domain-dependent, passing between Cohere rerank-v4.0-pro and ZeroEntropy zerank-2 by subset. The models therefore do not solely differ on aggregate. They expose different pieces of the candidate pool and recover different queries at different cutoffs, which is precisely the structure fusion exists to exploit. Shown in Table 5, the same structure reappears at the listwise stage with each model recovering queries the other two miss. GPT-5.6 Terra uniquely solves 28.3 queries on average, GPT-5.6 Luna 16.0, and GPT-5.4 Mini 10.3, for 54.6 model-unique rank-1 successes in total. Even the weakest singleton places a relevant document first on roughly ten queries both stronger models miss.

		Success@1		Recall@20	
Subset	n	c / v / z	any	c / v / z	any
Biology	102	0 / **5** / 3	8	**2** / 1 / 0	3
Earth Science	116	3 / **4** / 3	10	**1** / **1** / 0	2
Economics	103	1 / **6** / **6**	13	1 / 0 / **3**	4
Psychology	99	**7** / 3 / 3	13	**3** / 0 / 0	3
Robotics	92	1 / 0 / **7**	8	1 / 0 / **4**	5
Total	512	12 / 18 / **22**	52	**8** / 2 / 7	17

Table 4: Per-reranker unique successes at cutoffs 1 and 20, measured at the k=200 operating point where all three singletons are at or near their peak. In this table "c" denotes Cohere rerank-v4.0-pro, "v" denotes Voyage rerank-2.5, and "z", ZeroEntropy zerank-2. The “any” column is the sum of the three per-model counts, since solo wins are mutually exclusive by construction. The model with the most unique successes per subset is highlighted in bold.

Cutoff	**GPT-5.4 Mini only**	**GPT-5.6 Luna only**	**GPT-5.6 Terra only**	**Any unique**
Success@1	10.3	16.0	**28.3**	54.6

Table 5: Trial-aligned unique Success@1 cases among the three listwise rerankers over 512 queries, averaged across three trials. The model with the most unique successes is highlighted in bold.

3.5.2 Equal-Weight Fusion

The simplest way to exploit this disagreement is with static fusion. Combine every model's ranking with equal weights, for every query, with no training. Whatever such a rule achieves is a floor on the value of width, since it requires no learning, no tuning, and no per-query decisions. This is also appealing from a latency perspective. Since the models operate independently over the same candidate set, their calls may be issued in parallel at both stages, with latency bounded by the slowest member. Table 6 reports the equal-weight fusion of the three cross-encoders against the best singleton across the depth sweep, and Table 7 the equal-weight fusion of the three listwise models. At the cross-encoder stage, fusion is at best a marginal improvement. On the primary nDCG@10 metric, the stronger rule, RSF, matches ZeroEntropy zerank-2 at the shallowest pool, 0.362 against 0.358 at k=100, and pulls ahead only modestly as pools deepen, reaching 0.438 against 0.424 at k=2,000, while RRF stays within 0.002 of the singleton through k=200 and pulls ahead only from k=500. Success@1 is more generous to the blend, with RSF reaching 0.421 against the singleton's 0.393 at the deepest pool, but the graded metric shows those gains concentrate at a single rank position rather than reflecting a better ordering, and Recall@20 is essentially unchanged throughout, within 0.008 of the best singleton at every depth. At the listwise stage, the same rule fares worse. The equal-weight fusion of GPT-5.4 Mini, GPT-5.6 Luna, and GPT-5.6 Terra reaches 0.471 nDCG@10 and 0.483 Success@1, below the best singleton, GPT-5.6 Terra, on both metrics, 0.488 and 0.502, respectively.

We regard this as the section's main finding. Section 3.5.1 established that the models disagree on over fifty queries at k=200. However, equal-weight fusion converts almost none of that disagreement into ranking quality. A uniform rule must grant every member influence on every query to profit from the few it uniquely solves, and it pays for that influence everywhere else. The unconverted disagreement is the empirical case for future work on query-dependent weighting and routing [32].

	k=100	k=200	k=500	k=1,000	k=2,000
nDCG@10					
Best Singleton (ZeroEntropy zerank-2)	0.358	0.396	0.415	0.423	0.424
Equal-weight fusion, RRF	0.359	0.394	0.419	0.429	0.432
Equal-weight fusion, RSF	**0.362**	**0.400**	**0.424**	**0.438**	**0.438**
Success@1					
Best Singleton	0.354	0.383	0.393	0.397	0.393
Equal-weight fusion, RRF	0.373	0.390	0.408	0.410	0.412
Equal-weight fusion, RSF	**0.380**	**0.404**	**0.415**	**0.423**	**0.421**
Recall@20					
Best Singleton	0.418	0.480	0.528	0.555	0.560
Equal-weight fusion, RRF	**0.421**	**0.482**	**0.533**	0.559	0.562
Equal-weight fusion, RSF	**0.421**	0.479	0.530	**0.563**	**0.566**

Table 6: Equal-weight fusion of all three cross-encoders: Cohere rerank-v4.0-pro, Voyage rerank-2.5, ZeroEntropy zerank-2, against the best singleton, as means by first-stage depth. The best singleton at each depth and metric is the strongest individual reranker. Under the mean this is ZeroEntropy zerank-2 at every depth and metric. Bold marks the best condition per depth and metric.

Condition	Mean nDCG@10	Mean Success@1
ZeroEntropy zerank-2 top-20	0.424	0.393
GPT-5.4 Mini	0.424 ± 0.000	0.419 ± 0.004
GPT-5.6 Luna	0.470 ± 0.002	0.478 ± 0.004
GPT-5.6 Terra	$\mathbf{0.488 \pm 0.003}$	$\mathbf{0.502 \pm 0.004}$
Equal-weight RRF (Mini, Luna, Terra)	0.471 ± 0.004	0.483 ± 0.004

Table 7: Mean nDCG@10 and Success@1 across all five subsets for the ZeroEntropy zerank-2 input ordering, the three listwise rerankers, and the equal-weight fusion of all three. Recall@20 is fixed at the ZeroEntropy zerank-2 input value for every condition by construction (Section 2.4).

4 Discussion

4.1 The Cost of Depth, Width, and Stages

The preceding sections treat depth, stages, and width as levers on ranking quality. This section prices them, so that the ladder of Figure 1 can be read as a path along a cost–quality Pareto frontier rather than as a sequence of free choices. We model per-query inference cost from provider list prices as of July 31, 2026 under two assumptions: every candidate document contains $d = 500$ tokens and every query contains $q = 200$ tokens. All figures are undiscounted list prices for synchronous traffic. The complete price and rate-limit catalogue, with the billing formulas by which readers with different document or query lengths can rescale every number below, appears in Appendix F. The three cross-encoders bill under two conventions. Voyage rerank-2.5 and ZeroEntropy zerank-2 bill per processed token, with the query counted once per document scored, $(q + d) \cdot k = 700k$ billed tokens at depth k, at \$0.05 and \$0.025 per million tokens respectively, so per-query cost is linear in retrieval depth. Cohere bills \$0.0025 per search unit covering a query and up to 100 documents, but splits any document that exceeds 500 tokens once the query is included into chunks that each count as a document. Under our assumptions every document thus bills as two chunks and a query at depth k consumes $\lceil 2k/100 \rceil$ units. ZeroEntropy zerank-2 is therefore the cheapest model at every depth, Voyage rerank-2.5 twice its price, and Cohere rerank-v4.0-pro nearly three times. The listwise stage bills as ordinary LLM input and output, roughly 10,400 input tokens for a top-20 window plus roughly 100 output tokens for the returned permutation at reasoning effort `none`, pricing a GPT-5.6 Luna pass at \$0.0022 per query, a GPT-5.4 Mini pass at \$0.0083, and a GPT-5.6 Terra pass at \$0.0220. This prices the n=100 window tested in Section 3.4 with GPT-5.6 Luna at \$0.010. Beyond per-query price, rate limits constrain deep retrieval in practice. A single k=2,000 query consumes about 1.4M billed tokens at Voyage rerank-2.5, whose endpoint allows 2M tokens per minute at usage Tier 1 and 6M at Tier 3, so sustained throughput at full depth is one to four queries per minute per organization without a negotiated increase, and per-request caps at both

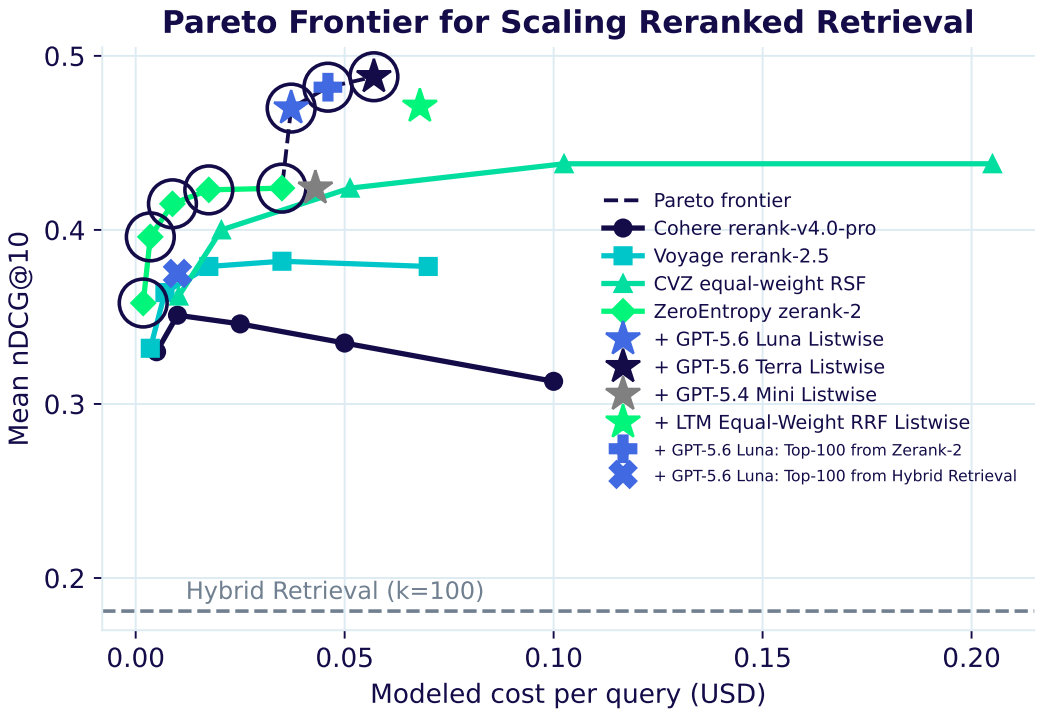


Figure 5: Pareto frontier for scaling reranked retrieval, plotting mean nDCG@10 across the five BRIGHT subsets against modeled per-query cost. Each cross-encoder series and the CVZ equal-weight RSF fusion contain five points, one per first-stage retrieval depth $k \in \{100, 200, 500, 1{,}000, 2{,}000\}$, with cost increasing in depth. Star markers are listwise passes over ZeroEntropy zerank-2's retained top-20 at k=2,000, including the equal-weight RRF fusion of all three listwise models (LTM: GPT-5.6 Luna, GPT-5.6 Terra, GPT-5.4 Mini). The two additional GPT-5.6 Luna conditions re-order a 100-document window, retained by ZeroEntropy zerank-2 at k=2,000 and taken directly from hybrid retrieval with no cross-encoder, described further in Section 4.2. Costs follow the billing conventions of Appendix F at list prices as of July 31, 2026, under d=500 and q=200, with first-stage retrieval amortized into the index. Circled points are Pareto-efficient: ZeroEntropy zerank-2 at every depth, followed by the GPT-5.6 Luna top-20 and top-100 passes and the GPT-5.6 Terra top-20 pass. The dashed horizontal line marks hybrid retrieval without reranking. No fusion condition reaches the frontier.

token-billed providers force each deep query to be split across multiple API calls. Cohere's production limit of 1,000 requests per minute is comparatively more enabling.

Figure 5 plots every end-to-end condition of the study in this cost model against the ordering quality it delivers, and the Pareto-efficient set traces the paper's ladder exactly: ZeroEntropy zerank-2 at all five depths, then the GPT-5.6 Luna and GPT-5.6 Terra listwise passes. The frontier's shape prices each lever. Along the depth rungs the marginal price of ordering climbs steeply: roughly \$0.0004 per 0.01 of mean nDCG@10 from k=100 to k=200, \$0.003 from k=200 to k=500, and \$0.011 from k=500 to k=1,000. The final rung into k=2,000 then costs \$0.0175 for a gain of 0.001, effectively buying no ordering at all. The Luna pass then buys the tallest vertical step along the frontier, 0.046 nDCG@10 and 0.085 Success@1 for \$0.0022, about \$0.0005 per point of nDCG@10, paid at the top of the ladder where depth has nothing left to sell. GPT-5.4 Mini costs four times Luna's price and leaves mean nDCG@10 unchanged, and GPT-5.6 Terra costs ten times Luna's price for a further 0.018. Assembled end-to-end, the frontier's recommended configuration costs \$0.0372 per query, \$0.035 for ZeroEntropy zerank-2 over the k=2,000 pool and \$0.0022 for the Luna pass, and carries mean nDCG@10 from 0.181 to 0.470. In this pipeline, the cross-encoder consumes 94% of the spend and buys reach, distilling a pool with recall 0.871 into twenty documents with recall 0.560. The listwise stage then consumes the remaining 6% and buys the pipeline's largest ordering gain, shown in Table 8. Width is the lever the frontier leaves behind. The equal-weight three-way cross-encoder fusion sums its members' prices to \$0.205 per query at k=2,000, nearly six times the singleton and the most expensive condition in the study for cross-subset gains of 0.014 nDCG@10 and 0.028 Success@1. The three-model listwise ensemble adds \$0.0325 per query, fifteen times Luna's price, and finishes below the best singleton on both ordering metrics. Both fusion conditions hang visibly off the frontier. The disagreement width targets is real, Section 3.5.1 measures roughly fifty model-unique rank-1 successes,

Condition	Cost/query	Mean S@1	Mean nDCG@10
Hybrid retrieval	\$0.000	0.153	0.181
ZeroEntropy zerank-2 (singleton)	\$0.035	0.393	0.424
Voyage rerank-2.5 (singleton)	\$0.070	0.374	0.379
Cohere rerank-v4.0-pro (singleton)	\$0.100	0.247	0.313
CVZ equal-weight RSF	\$0.205	0.421	0.438
ZeroEntropy zerank-2 + GPT-5.4 Mini (top-20)	\$0.043	0.419 ± 0.004	0.424 ± 0.000
ZeroEntropy zerank-2 + GPT-5.6 Luna (top-20)	\$0.037	0.478 ± 0.004	0.470 ± 0.002
ZeroEntropy zerank-2 + GPT-5.6 Terra (top-20)	\$0.057	0.502 ± 0.004	0.488 ± 0.003
ZeroEntropy zerank-2 + Mini+Luna+Terra RRF	\$0.068	0.483 ± 0.004	0.471 ± 0.004
ZeroEntropy zerank-2 + GPT-5.6 Luna (top-100)	\$0.045	0.486 ± 0.008	0.482 ± 0.003
Hybrid retrieval (k=100) + GPT-5.6 Luna (top-100)	\$0.010	0.392 ± 0.003	0.375 ± 0.002

Table 8: Cost versus performance at k=2,000, complementing the ordering view of Figure 5 with Success@1 and nDCG@10. Per-query cost counts reranking inference only. First-stage retrieval is amortized into the index and enters every row at zero, so the unreranked hybrid baseline, whose top-20 is identical at every depth under the nested-prefix sweep, prices at \$0.000. The final block prices the two 100-document listwise windows of Section 4.2. The cross-encoder free condition re-orders the hybrid top-100 at k=100. Listwise rows are means across three trials $\pm$ across-trial standard deviation. Costs are modeled from list prices as of July 31, 2026 under d=500 and q=200 token length assumptions. See Appendix F for billing formulas.

but equal-weight fusion converts almost none of it while paying every member's price on every query. At these rates, static width buys insurance against a drowning member rather than quality, and the economical form of width is the query-dependent routing this paper leaves to future work, which has the potential to pay for one model's judgment per query rather than three.

Depth may have an economical form as well. Our cost model takes list prices as given, and the cross-encoder's 94% share reflects the depth it reaches: each of its k=2,000 candidates is scored by a full joint forward pass over the query and the document, so the document's d tokens, five-sevenths of the bill under our assumptions, are processed anew on every query. Architectures that need less of that computation are one route by which the price of reach could fall. Our pipeline tests only the two ends of the interaction spectrum, offline representations at the first stage and online joint encoding at the second, and the space between them is not empty. Late-interaction models such as ColBERT [19, 40] precompute token-level document representations and interact with the query through MaxSim. MICE [51], a minimal-interaction cross-encoder, encodes documents once offline and runs only the query tokens through a few upper layers that cross-attend to the cached document. Its authors report up to $2.5\times$ fewer FLOPs and roughly $4\times$ lower latency with precomputation at a small cost in in-domain effectiveness. Whether such savings reach list price is a provider's decision, but a reranker that passed them on would shift every depth rung of Figure 5 to the left. Like query-dependent routing, that is an extension this study leaves open.

4.2 Cutting out the Cross-Encoder Stage

Deepening the listwise window multiplies the listwise outlay without scaling its return, and Figure 5 shows the trade directly as the near-horizontal step between the two GPT-5.6 Luna passes on the frontier. The n=100 pass of Section 3.4 bills roughly 50,700 input tokens, about \$0.010 per query with Luna, five times the top-20 pass, and buys a gain of only 0.012 nDCG@10 over that pass while improving retention only marginally, 0.565 Recall@20 against the input ordering's 0.560. The deeper window is therefore Pareto-efficient in the narrow sense that nothing cheaper matches its ordering, but five times the spend for marginal movement on one metric and none on the other keeps the top-20 window as the ladder's operating point, an economic conclusion as much as a quality one.

The same n=100 protocol prices a second question: *do we still need a cross-encoder stage when the listwise reranker reads deeper windows?* Table 9 holds the window length fixed and varies only the stage that produced it. Handed hybrid retrieval's top-100, Luna reaches 0.375 mean nDCG@10, 0.392 Success@1, and 0.421 Recall@20 for its \$0.010 per query. ZeroEntropy zerank-2 over that same k=100 pool reaches 0.358, 0.354, and 0.418 for \$0.0018. Over an identical shallow pool, the listwise model matches the cross-encoder's retention and exceeds its ordering on both metrics, at six times the price. What separates the two windows in Table 9, however, is not the ordering model but the availability each carries. ZeroEntropy zerank-2's retained top-100, distilled from a k=2,000 pool, holds 0.738 mean

Condition	nDCG@10	Success@1	Recall@20
Input: hybrid top-100 (window recall 0.476)			
Hybrid input ordering	0.181	0.153	0.282
ZeroEntropy zerank-2 over the window (k=100)	0.358	0.354	0.418
Luna over the window	0.375 $\pm$ 0.002	0.392 $\pm$ 0.003	0.421 $\pm$ 0.001
Input: ZeroEntropy zerank-2 retained top-100 at k=2,000 (window recall 0.738)			
ZeroEntropy zerank-2 input ordering	0.424	0.393	0.560
Luna over the window	**0.482** $\pm$ 0.003	**0.486** $\pm$ 0.008	**0.565** $\pm$ 0.004

Table 9: Removing the cross-encoder. Both panels hand a listwise model a 100-document window; only the stage that produced the window differs. The parenthetical window recall is the mean availability ceiling each window carries, from Table 30 and Table 10. Luna rows are means across three trials $\pm$ across-trial standard deviation; deterministic rows carry none. Bold marks the best value per metric across the table. Per-subset results appear in Tables 30 and 31.

recall against the hybrid window's 0.476, and Luna over it reaches 0.482 nDCG@10 against 0.375 over the hybrid window at identical listwise cost, a gap of 0.107 that no amount of re-ordering the shallower window can close. In Figure 5 the cross-encoder-free condition accordingly sits well below the frontier despite being the cheapest listwise configuration in the study. What the cross-encoder sells at this stage is reach rather than ordering, and no listwise budget substitutes for it. Gollapudi et al. [15] reach the same conclusion from the opposite end of the design space. A single language model that reads the entire corpus in context and decodes the relevant document's identifier collapses near a million tokens, and the configuration that recovers dense-retrieval accuracy prunes the pool to a top-B shortlist at an intermediate layer, reinstating the stage the design was meant to remove.

5 Related Work

5.1 Scaling Candidate Depth

Candidate depth determines both the evidence available to a reranker and the computation required to score it. Jacob et al. study how cross-encoder effectiveness changes as this depth increases, finding that rerankers initially benefit from larger candidate pools but eventually saturate or regress, a phenomenon they term *Drowning in Documents* [18]. They attribute this behavior to phantom hits, irrelevant candidates that receive spuriously high relevance scores and displace relevant documents near the head of the ranking. Deeper pools therefore create two competing effects. They make more relevant documents available to the reranker, but also expose it to more opportunities to promote high-scoring distractors. We observe the same depth-dependent tension in our experiments. Cohere rerank-v4.0-pro regresses at Success@1 as k grows from 100 to 2,000, while Voyage rerank-2.5 and ZeroEntropy zerank-2 stop improving at the head of the ranking. At the same time, Recall@20 continues to increase for the strongest reranker. Deeper pools can therefore improve retention even after their contribution to the final ordering has saturated. Tchuindjo et al. [48] frame this trade-off as an asymmetry between retrieval and verification and place BRIGHT among the benchmarks whose gold documents are retrievable into a deep pool but hard to order, in contrast to their oblique tasks where deeper retrieval leaves the availability ceiling near zero.

The aggregate saturation and regression observed across queries conceal substantial query-level variation. Huo et al. directly measure per-query depth effectiveness using MonoT5 and MonoELECTRA over MSMARCO and TREC DL 2019 [17]. They find that some queries improve with deeper reranking, some plateau early, and others deteriorate below their initial ranking. They consequently formulate reranking depth as a query-specific prediction problem. An LLM-assessed estimate of first-stage ranking quality reduces average reranking cost by a factor of three while preserving effectiveness. They also show that oracle depth selection improves effectiveness at an average depth approximately five times smaller. This query-adaptive view has an earlier precedent in dynamic cutoff prediction. Culpepper et al. predict a query-specific cutoff k, selecting how many first-stage candidates to pass to downstream ranking in order to improve efficiency within a prescribed effectiveness envelope [9]. Reranker-Guided Search addresses the complementary question of which candidates to score under a fixed reranking budget. Rather than reranking a sequential prefix, it uses reranker feedback to guide a search over a document-proximity graph [57]. Together, these works show that both the number and selection of candidates presented to a reranker can be treated as allocation decisions rather than fixed globally. Our per-query analysis addresses a different allocation question: whether heterogeneous rerankers succeed on different queries and whether equal-weight fusion can exploit their complementary errors. Depth itself remains globally controlled in our experiments.

5.2 Composing Ranking Stages

Multi-stage ranking systems balance effectiveness and efficiency by applying increasingly expensive ranking functions to progressively smaller candidate sets. Wang et al. formalized this design as a cascade that repeatedly prunes and refines the candidate set as it moves through the pipeline [54]. Transformer-based reranking instantiates the same pattern with more expressive interaction models. Nogueira and Cho apply BERT as a cross-encoder that jointly represents a query and candidate passage [29], while Nogueira et al. compose pointwise MonoBERT and pairwise DuoBERT within a multi-stage architecture whose quality–latency trade-off is controlled by the number of candidates admitted to each stage [30]. The Expando-Mono-Duo pattern similarly composes first-stage document expansion, pointwise MonoT5 reranking, and pairwise DuoT5 reranking [34]. These systems establish the central logic of ranking cascades: later stages spend more computation on richer comparisons, but may act only on the candidates retained by the stages before them.

Within such cascades, reranking methods differ in how candidates are compared. Pointwise models score each document against the query in isolation, while pairwise models directly compare two candidates. Listwise methods instead consider several candidates jointly and return an ordering over the set. RankGPT demonstrates that prompted generative LLMs can act as zero-shot listwise rerankers, directly reordering a supplied candidate list [47]. RankZephyr distills this listwise capability into an open model and applies it to longer rankings through progressive sliding windows [36]. Setwise prompting occupies an intermediate design point, asking a reranker to select among small groups of candidates in order to reduce the inference cost of pairwise comparison without requiring a full listwise context [59]. Our pipeline composes two of these comparison structures, with a pointwise commercial cross-encoder constructing a retained shortlist that a listwise LLM may subsequently reorder.

5.3 Widening Through Fusion

Combining the outputs of multiple retrieval systems is a long-studied problem in information retrieval. Fox and Shaw introduced the CombSUM, CombMNZ, CombMAX, and related operators for aggregating relevance evidence across independent retrieval runs [11]. Lee subsequently showed that fusion is most effective when systems retrieve similar relevant documents but different non-relevant documents, so that shared evidence is reinforced while model-specific errors are diluted [22]. IRPAPERS demonstrates this complementarity across retrieval modalities. Text-based and image-based systems recover different scientific document pages, and their fusion outperforms either modality alone [44]. Score-based fusion requires the outputs of heterogeneous systems to be made comparable, motivating the normalization methods studied by Montague and Aslam [28]. Renda and Straccia compare these score-based methods with rank-based aggregation [39], while Reciprocal Rank Fusion discards score magnitude entirely and combines systems according to the ranks they assign [8]. These two traditions correspond directly to the Relative Score Fusion and Reciprocal Rank Fusion methods we evaluate. Whereas this literature primarily combines independent first-stage retrieval systems or representations, we apply the same methods to heterogeneous rerankers scoring a shared candidate pool.

Fusion has also been applied directly to neural ranking models. Gao et al. ensemble BERT, RoBERTa, and ELECTRA rerankers over a shared top-100 candidate pool [12]. Gecko combines rankings produced by query-likelihood and relevance classification LLM prompts through RRF. In their work, the fused judgments are used to construct training data for an embedding model rather than as a deployed reranking stage [21]. CAME moves beyond independently trained models by jointly learning lexical, local, and global retrieval experts and encouraging each to specialize in different relevance patterns [16]. These approaches exploit diversity through either static fusion or jointly learned specialization, while Adaptive Re-Ranking instead treats model choice as a query-dependent routing problem, selecting among ranking strategies of different cost and capability [14]. Our work differs in that we combine frozen commercial cross-encoders and listwise LLMs at inference time, without additional training or tuning. We then measure whether these heterogeneous models succeed on different queries and whether equal-weight fusion can convert their complementary errors into ranking quality.

5.4 Test-Time Compute for Search

Recent work studies test-time scaling through repeated sampling, adaptive compute allocation, and compound AI systems [45, 2, 5]. Search broadens this allocation problem because compute may be spent before retrieval on formulating the information need, during retrieval on expanding or exploring the candidate set, or afterward on evaluating and ordering results. Query-writing methods spend this budget upstream. HyDE generates a hypothetical relevant document and retrieves using its embedding [13], Query2doc augments the query with generated pseudo-documents [53], and Rewrite–Retrieve–Read rewrites the input for a frozen retriever and downstream reader [25]. Generalized Pseudo-Relevance Feedback further incorporates retrieved documents into model-independent natural-

language rewrites while training for robustness to irrelevant feedback [50]. Iterative systems extend this strategy across multiple rounds by interleaving reformulation, retrieval, and reasoning [49, 1]. Beyond free-text reformulation, Shorten et al. study LLMs formulating structured database queries directly through Function Calling, unifying search with filters and aggregations [43]. At the reranking stage, Rank1 allocates additional inference compute to pointwise relevance reasoning, while Rank-K extends test-time reasoning to listwise reranking [55, 58]. These approaches complement the system-level axes studied in our work, showing that additional search compute may be spent on representing the information need, exploring the corpus, reasoning within a reranker, or composing multiple ranking operations.

6 Conclusion

This work studies how inference-time compute should be allocated across retrieval depth, additional ranking stages, and model width within a stage. Read as a single pipeline, the stages compose. Hybrid retrieval alone returns a top-20 with a mean 0.181 nDCG@10 and 0.282 Recall@20, a single cross-encoder over a k=100 pool reaches 0.358 nDCG@10 and 0.418 Recall@20. Deepening retrieval to k=2,000 lifts the reranked list to 0.424 and 0.560, and a listwise pass lifts nDCG@10 a further 0.046, from 0.424 to 0.470. Each stage purchases a distinct, measurable gain, and this ladder is the practical recipe this work documents. The rungs are not interchangeable, however, because each acts on a different failure mode. Increasing hybrid retrieval depth from k=100 to k=2,000 raises mean pool recall from 0.476 to 0.871, and the cross-encoders convert this expanded ceiling asymmetrically. We find that nDCG@10 rises from 0.358 to 0.423 at k=1,000 and then saturates at 0.424, while Recall@20 continues to improve, reproducing the central tension described by *Drowning in Documents*. The listwise stage then resolves ordering within the retained set, and its effectiveness depends strongly on the model performing the joint comparison. GPT-5.4 Mini fails to improve the ZeroEntropy zerank-2 ordering, matching it at 0.424, while GPT-5.6 Luna and GPT-5.6 Terra reach 0.470 and 0.488, respectively.

Even the strongest ordering methods remain below the capture ceiling. Increasing the reranked output cutoff from 20 to 100 raises mean recall from 0.560 to 0.738, but a substantial gap remains below the 0.871 first-stage ceiling. These documents were found by retrieval and could fit within the expanded output, yet the rerankers still left them beneath too many distractors. Adding depth to the listwise stage to reach them does not resolve the residual. Over ZeroEntropy zerank-2's retained top-100, GPT-5.6 Luna gains only 0.012 mean nDCG@10 over the top-20 pass at five times the cost, and recovers retained-set recall only marginally. The lever beyond this ladder is width, and our result there is deliberately negative. Heterogeneous models recover different queries at both reranking stages: Cohere rerank-v4.0-pro, Voyage rerank-2.5, and ZeroEntropy zerank-2 contribute 52 model-unique successes at rank 1 across 512 queries, and the listwise models tested, GPT-5.4 Mini, GPT-5.6 Luna, and GPT-5.6 Terra, uniquely solve an average of 10.3, 16.0, and 28.3, respectively. Yet training-free equal-weight fusion converts almost none of this disagreement. At the cross-encoder stage the blend matches the best singleton at shallow depth and exceeds it only marginally at the deepest pools, with its Success@1 gains concentrated at a single rank position. At the listwise stage, the blend falls short of the strongest singleton on both ordering metrics. A uniform rule must grant every member influence on every query to profit from the few it uniquely recovers, and pays for that influence everywhere else. Static fusion therefore establishes a training-free floor that sits at or near the best singleton, and converting the documented disagreement into ranking quality appears to require query-dependent weighting or routing. A cascade in which fused cross-encoders construct the retained set, and learned routers at both stages, are the natural next experiments. The results support a stage-aware view of inference-time scaling. Retrieval depth determines which relevant documents remain available, stages resolve the ordering contests that remain, and the models' complementary errors mark routing, rather than static blending, as the most promising lever left unexploited.

References

[1] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. In *Proceedings of the 12th International Conference on Learning Representations*, 2024.

[2] Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large Language Monkeys: Scaling Inference Compute with Repeated Sampling. *arXiv preprint arXiv:2407.21787*, 2024.

[3] Sebastian Bruch, Siyu Gai, and Amir Ingber. An analysis of fusion functions for hybrid retrieval. *ACM Transactions on Information Systems*, 42(1):1–35, 2023.

[4] Jianlyu Chen, Junwei Lan, Chaofan Li, Defu Lian, and Zheng Liu. ReasonEmbed: Enhanced text embeddings for reasoning-intensive document retrieval. In *Proceedings of the 64th Annual Meeting of the Association for*

Computational Linguistics (Volume 1: Long Papers), pages 1203–1221, San Diego, California, United States, July 2026. Association for Computational Linguistics.

[5] Lingjiao Chen, Jared Quincy Davis, Boris Hanin, Peter Bailis, Ion Stoica, Matei Zaharia, and James Zou. Are More LLM Calls All You Need? Towards Scaling Laws of Compound Inference Systems. In *Advances in Neural Information Processing Systems*, 2024.

[6] Ruey-Cheng Chen, Luke Gallagher, Roi Blanco, and J. Shane Culpepper. Efficient cost-aware cascade ranking in multi-stage retrieval. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 445–454. ACM, 2017.

[7] Cohere. Rerank v4.0. `https://docs.cohere.com/docs/rerank-overview`. Accessed July 2026.

[8] Gordon V. Cormack, Charles L. A. Clarke, and Stefan Büttcher. Reciprocal Rank Fusion Outperforms Condorcet and Individual Rank Learning Methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 758–759. ACM, 2009.

[9] J. Shane Culpepper, Charles L. A. Clarke, and Jimmy Lin. Dynamic cutoff prediction in multi-stage retrieval systems. In *Proceedings of the 21st Australasian Document Computing Symposium*, pages 17–24. ACM, 2016.

[10] Etienne Dilocker, Bob van Luijt, Byron Voorbach, Mohd Shukri Hasan, Abdel Rodriguez, Dirk Alexander Kulawiak, Marcin Antas, and Parker Duckworth. Weaviate. `https://github.com/weaviate/weaviate`, Accessed August 2026.

[11] Edward A. Fox and Joseph A. Shaw. Combination of Multiple Searches. In *Proceedings of the Second Text REtrieval Conference (TREC-2)*, volume 500-215 of *NIST Special Publication*, pages 243–252. National Institute of Standards and Technology, 1994.

[12] Luyu Gao, Zhuyun Dai, and Jamie Callan. Rethink Training of BERT Rerankers in Multi-Stage Retrieval Pipeline. In *Advances in Information Retrieval*, pages 280–286. Springer, 2021.

[13] Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 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, Toronto, Canada, 2023. Association for Computational Linguistics.

[14] Ata Cinar Genc, Emir Kaan Korukluoglu, and James Allan. Adaptive re-ranking. In *Proceedings of the 2026 International ACM SIGIR Conference on Innovative Concepts and Theories in Information Retrieval*, New York, NY, USA, July 2026. Association for Computing Machinery.

[15] Siddharth Gollapudi, Nilesh Gupta, Prasann Singhal, and Sewon Min. Can language models actually retrieve in-context? drowning in documents at million token scale. *arXiv preprint arXiv:2607.01538*, 2026.

[16] Jiafeng Guo, Yinqiong Cai, Keping Bi, Yixing Fan, Wei Chen, Ruqing Zhang, and Xueqi Cheng. CAME: Competitively Learning a Mixture-of-Experts Model for First-Stage Retrieval. *ACM Transactions on Information Systems*, 43(2):1–25, 2025.

[17] Siqing Huo, Andrew Parry, Debasis Ganguly, and Charles L. A. Clarke. Optimal re-ranking depth. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '26, pages 3799–3804, New York, NY, USA, 2026. Association for Computing Machinery.

[18] Mathew Jacob, Erik Lindgren, Matei Zaharia, Michael Carbin, Omar Khattab, and Andrew Drozdov. Drowning in Documents: Consequences of Scaling Reranker Inference. *arXiv preprint arXiv:2411.11767*, 2024.

[19] Omar Khattab and Matei Zaharia. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20)*, 2020.

[20] Carlos Lassance, Hervé Déjean, Thibault Formal, and Stéphane Clinchant. SPLADE-v3: New baselines for SPLADE. *arXiv preprint arXiv:2403.06789*, 2024.

[21] Jinhyuk Lee, Zhuyun Dai, Xiaoqi Ren, Blair Chen, Daniel Cer, Jeremy R. Cole, Kai Hui, Michael Boratko, Rajvi Kapadia, Wen Ding, Yi Luan, Sai Meher Karthik Duddu, Gustavo Hernández Ábrego, Weiqiang Shi, Nithi Gupta, Aditya Kusupati, Prateek Jain, Siddhartha Reddy Jonnalagadda, Ming-Wei Chang, and Iftekhar Naim. Gecko: Versatile Text Embeddings Distilled from Large Language Models. *arXiv preprint arXiv:2403.20327*, 2024.

[22] Joon Ho Lee. Analyses of Multiple Evidence Combination. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 267–276. ACM, 1997.

[23] Wenhan Liu, Xinyu Ma, Weiwei Sun, Yutao Zhu, Yuchen Li, Dawei Yin, and Zhicheng Dou. 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, United States, July 2026. Association for Computational Linguistics.

[24] Meixiu Long, Duolin Sun, Dan Yang, Junjie Wang, Yue Shen, Jian Wang, Peng Wei, Jinjie Gu, and Jiahai Wang. DIVER: A multi-stage approach for reasoning-intensive information retrieval. *arXiv preprint arXiv:2508.07995*, 2025.

[25] Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. Query rewriting in retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5303–5315, Singapore, 2023. Association for Computational Linguistics.

[26] Xueguang Ma, Xinyu Zhang, Ronak Pradeep, and Jimmy Lin. Zero-shot listwise document reranking with a large language model. *arXiv preprint arXiv:2305.02156*, 2023.

[27] Luke Merrick, Danmei Xu, Gaurav Nuti, and Daniel Campos. Arctic-embed: Scalable, efficient, and accurate text embedding models. *arXiv preprint arXiv:2405.05374*, 2024.

[28] Mark Montague and Javed A. Aslam. Relevance Score Normalization for Metasearch. In *Proceedings of the 10th International Conference on Information and Knowledge Management*, pages 427–433. ACM, 2001.

[29] Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with BERT. *arXiv preprint arXiv:1901.04085*, 2019.

[30] Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. Multi-stage document ranking with BERT. *arXiv preprint arXiv:1910.14424*, 2019.

[31] OpenAI. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

[32] Melissa Z. Pan, Negar Arabzadeh, Mathew Jacob, Fiodar Kazhamiaka, Esha Choukse, and Matei Zaharia. Natural language query to configuration for retrieval agents. *arXiv preprint arXiv:2605.27361*, 2026.

[33] Nicholas Pipitone, Ghita Houir Alami, Advaith Avadhanam, Anton Kaminskyi, and Ashley Khoo. zELO: ELO-inspired training method for rerankers and embedding models. *arXiv preprint arXiv:2509.12541*, 2025.

[34] Ronak Pradeep, Rodrigo Nogueira, and Jimmy Lin. The expando–mono–duo design pattern for text ranking with pretrained sequence-to-sequence models. *arXiv preprint arXiv:2101.05667*, 2021.

[35] Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. RankVicuna: Zero-shot listwise document reranking with open-source large language models. *arXiv preprint arXiv:2309.15088*, 2023.

[36] Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. RankZephyr: Effective and Robust Zero-Shot Listwise Reranking is a Breeze! *arXiv preprint arXiv:2312.02724*, 2023.

[37] Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, Xuanhui Wang, and Michael Bendersky. Large language models are effective text rankers with pairwise ranking prompting. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1504–1518. Association for Computational Linguistics, 2024.

[38] Qwen Team. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[39] M. Elena Renda and Umberto Straccia. Web Metasearch: Rank vs. Score Based Rank Aggregation Methods. In *Proceedings of the 2003 ACM Symposium on Applied Computing*, pages 841–846. ACM, 2003.

[40] Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2022)*, 2022.

[41] Sahel Sharifymoghaddam, Yijun Ge, Raghav Vasudeva, and Jimmy Lin. Lighting the way for BRIGHT: Reproducible baselines with anserini, pyserini, and RankLLM. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2026. arXiv:2509.02558.

[42] Connor Shorten, Charles Pierse, Thomas Benjamin Smith, Erika Cardenas, Akanksha Sharma, John Trengrove, and Bob van Luijt. StructuredRAG: JSON response formatting with large language models. *arXiv preprint arXiv:2408.11061*, 2024.

[43] Connor Shorten, Charles Pierse, Thomas Benjamin Smith, Karel D'Oosterlinck, Tuana Celik, Erika Cardenas, Leonie Monigatti, Mohd Shukri Hasan, Edward Schmuhl, Daniel Williams, Aravind Kesiraju, and Bob van Luijt. Querying databases with function calling. *arXiv preprint arXiv:2502.00032*, 2025.

[44] Connor Shorten, Augustas Skaburskas, Daniel M. Jones, Charles Pierse, Roberto Esposito, John Trengrove, Etienne Dilocker, and Bob van Luijt. IRPAPERS: A visual document benchmark for scientific retrieval and question answering. *arXiv preprint arXiv:2602.17687*, 2026.

[45] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM Test-Time Compute Optimally Can Be More Effective Than Scaling Model Parameters. In *Proceedings of the 13th International Conference on Learning Representations*, 2025.

[46] 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, Sercan Ö. Arik, Danqi Chen, and Tao Yu. BRIGHT: A Realistic and Challenging Benchmark for Reasoning-Intensive Retrieval. In *Proceedings of the 13th International Conference on Learning Representations*, 2025.

[47] Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. Is ChatGPT Good at Search? Investigating Large Language Models as Re-Ranking Agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14918–14937. Association for Computational Linguistics, 2023.

[48] Diane Tchuindjo, Devavrat Shah, and Omar Khattab. OBLIQ-Bench: Exposing overlooked bottlenecks in modern retrievers with latent and implicit queries. *arXiv preprint arXiv:2605.06235*, 2026.

[49] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving Retrieval with Chain-of-Thought Reasoning for Knowledge-Intensive Multi-Step Questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, pages 10014–10037. Association for Computational Linguistics, 2023.

[50] Yiteng Tu, Weihang Su, Yujia Zhou, Yiqun Liu, Fen Lin, Qin Liu, and Qingyao Ai. Generalized pseudo-relevance feedback. *arXiv preprint arXiv:2510.25488*, 2025.

[51] Mathias Vast, Victor Morand, Basile van Cooten, Laure Soulier, Josiane Mothe, and Benjamin Piwowarski. MICE: Minimal interaction cross-encoders for efficient re-ranking. *arXiv preprint arXiv:2602.16299*, 2026.

[52] Voyage AI. rerank-2.5. `https://docs.voyageai.com/docs/reranker`. Accessed July 2026.

[53] Liang Wang, Nan Yang, and Furu Wei. Query2doc: Query expansion with large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9414–9423, Singapore, 2023. Association for Computational Linguistics.

[54] Lidan Wang, Jimmy Lin, and Donald Metzler. A cascade ranking model for efficient ranked retrieval. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 105–114. ACM, 2011.

[55] Orion Weller, Kathryn Ricci, Eugene Yang, Andrew Yates, Dawn Lawrie, and Benjamin Van Durme. Rank1: Test-Time Compute for Reranking in Information Retrieval. In *Proceedings of the Conference on Language Modeling*, 2025.

[56] Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. C-Pack: Packed resources for general Chinese embeddings. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 641–649. Association for Computing Machinery, 2024.

[57] Haike Xu and Tong Chen. Beyond sequential reranking: Reranker-guided search improves reasoning intensive retrieval. *arXiv preprint arXiv:2509.07163*, 2025.

[58] Eugene Yang, Andrew Yates, Kathryn Ricci, Orion Weller, Vivek Chari, Benjamin Van Durme, and Dawn Lawrie. Rank-k: Test-time reasoning for listwise reranking. *arXiv preprint arXiv:2505.14432*, 2025.

[59] Shengyao Zhuang, Honglei Zhuang, Bevan Koopman, and Guido Zuccon. 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*, pages 38–47. ACM, 2024.

A Scaling Hybrid Retrieval

A.1 Recall by Subset and Depth

Subset	@1	@5	@20	@50	@100	@200	@500	@1000	@2000
Biology	0.137	0.110	0.218	0.328	0.391	0.509	0.698	0.856	0.907
Earth Science	0.207	0.227	0.353	0.471	0.551	0.622	0.714	0.789	0.830
Economics	0.117	0.141	0.256	0.378	0.494	0.613	0.759	0.849	0.886
Psychology	0.172	0.188	0.338	0.439	0.519	0.636	0.749	0.851	0.902
Robotics	0.130	0.106	0.247	0.321	0.423	0.564	0.708	0.762	0.829

Table 10: Hybrid retrieval recall by subset and depth. The @1 column reports Success@1 and all deeper cutoffs report pool Recall@k.

A.2 Native-Limit Retrieval Stability

Subset	Q	k=100	k=200	k=500	k=1,000	k=2,000
Success@1						
Biology	103	0.155 (0)	0.146 (0)	0.146 (0)	0.136 (0)	0.136 (0)
Earth Science	116	0.198 (0)	0.198 (0)	0.207 (0)	0.207 (0)	0.207 (0)
Economics	103	0.107 (0)	0.117 (0)	0.117 (0)	0.117 (0)	0.117 (0)
Psychology	101	0.178 (0)	0.178 (0)	0.178 (0)	0.188 (0)	0.188 (0)
Robotics	101	0.168 (0)	0.168 (0)	0.168 (0)	0.168 (0)	0.168 (0)
Recall@20						
Biology	103	0.199 (0)	0.202 (0)	0.209 (0)	0.214 (0)	0.220 (0)
Earth Science	116	0.338 ($\pm$0.001)	0.338 ($\pm$0.001)	0.344 ($\pm$0.001)	0.351 ($\pm$0.000)	0.353 ($\pm$0.001)
Economics	103	0.260 (0)	0.258 (0)	0.266 (0)	0.256 (0)	0.256 (0)
Psychology	101	0.329 (0)	0.332 (0)	0.339 (0)	0.340 (0)	0.347 (0)
Robotics	101	0.278 (0)	0.277 (0)	0.275 (0)	0.274 (0)	0.273 (0)

Table 11: Across-trial Success@1 (top panel) and Recall@20 (bottom panel) by native retrieval depth, for hybrid retrieval without reranking. Each (subset, k) is a fresh retrieval issued at `limit=k` and re-run five times, in contrast to the nested-prefix sweep of Section 2.3 used for all primary results. Cells are mean ($\pm$std) with (0) denoting identical results across all five trials, and ($\pm$0.000) nonzero variation that rounds to zero at three decimals. Variation appears only in Earth Science's Recall@20 and bounds the first-stage noise floor at roughly 0.001 (Section 3.1). This robustness check predates the cross-encoder stage and is computed over each subset's full query set rather than the intersection.

B Singleton Reranker Results

For the sake of visualization, the Success@1 and Recall@20 tables additionally include the recall ceiling the rerankers inherit for each subset and level of depth. First-stage pool Recall@K measures the fraction of judged relevant documents reachable after retrieval and is a direct ceiling on what any later stage can retain: a document absent from the pool cannot be recovered. We prefer this to first-stage Success@K because Success@K saturates once any one relevant document enters the pool. In the Success@1 tables the row is a reference rather than a bound, since a query with several gold documents can place one of them first while the pool holds only a fraction of the rest.

B.1 Success@1

Reranker	k=100	k=200	k=500	k=1,000	k=2,000
Hybrid Retrieval Ceiling	0.391	0.509	0.698	0.856	0.907
Cohere rerank-v4.0-pro	0.304	0.294	0.324	0.314	0.304
Voyage rerank-2.5	0.353	0.422	0.461	0.441	0.441
ZeroEntropy zerank-2	0.304	0.382	0.451	0.451	0.461

Table 12: Biology subset, Success@1.

Reranker	k=100	k=200	k=500	k=1,000	k=2,000
Hybrid Retrieval Ceiling	0.551	0.622	0.714	0.789	0.830
Cohere rerank-v4.0-pro	0.457	0.483	0.474	0.483	0.483
Voyage rerank-2.5	0.509	0.517	0.534	0.543	0.543
ZeroEntropy zerank-2	0.526	0.543	0.560	0.560	0.552

Table 13: Earth Science subset, Success@1.

Reranker	k=100	k=200	k=500	k=1,000	k=2,000
Hybrid Retrieval Ceiling	0.494	0.613	0.759	0.849	0.886
Cohere rerank-v4.0-pro	0.233	0.214	0.146	0.126	0.097
Voyage rerank-2.5	0.282	0.282	0.282	0.282	0.282
ZeroEntropy zerank-2	0.291	0.291	0.272	0.272	0.262

Table 14: Economics subset, Success@1.

Reranker	k=100	k=200	k=500	k=1,000	k=2,000
Hybrid Retrieval Ceiling	0.519	0.636	0.749	0.851	0.902
Cohere rerank-v4.0-pro	0.394	0.414	0.343	0.263	0.212
Voyage rerank-2.5	0.343	0.364	0.343	0.343	0.343
ZeroEntropy zerank-2	0.364	0.374	0.354	0.374	0.364

Table 15: Psychology subset, Success@1.

Reranker	k=100	k=200	k=500	k=1,000	k=2,000
Hybrid Retrieval Ceiling	0.423	0.564	0.708	0.762	0.829
Cohere rerank-v4.0-pro	0.228	0.217	0.196	0.174	0.141
Voyage rerank-2.5	0.250	0.250	0.261	0.261	0.261
ZeroEntropy zerank-2	0.283	0.326	0.326	0.326	0.326

Table 16: Robotics subset, Success@1.

B.2 Recall@20

Reranker	k=100	k=200	k=500	k=1,000	k=2,000
Hybrid Retrieval Ceiling	0.391	0.509	0.698	0.856	0.907
Cohere rerank-v4.0-pro	0.369	0.467	0.575	0.599	0.603
Voyage rerank-2.5	0.367	0.465	0.541	0.561	0.567
ZeroEntropy zerank-2	0.368	0.468	0.566	0.620	0.652

Table 17: Biology subset, Recall@20.

Reranker	k=100	k=200	k=500	k=1,000	k=2,000
Hybrid Retrieval Ceiling	0.551	0.622	0.714	0.789	0.830
Cohere rerank-v4.0-pro	0.504	0.534	0.560	0.576	0.580
Voyage rerank-2.5	0.504	0.525	0.560	0.577	0.577
ZeroEntropy zerank-2	0.497	0.534	0.577	0.603	0.613

Table 18: Earth Science subset, Recall@20.

Reranker	k=100	k=200	k=500	k=1,000	k=2,000
Hybrid Retrieval Ceiling	0.494	0.613	0.759	0.849	0.886
Cohere rerank-v4.0-pro	0.344	0.374	0.402	0.416	0.390
Voyage rerank-2.5	0.349	0.378	0.390	0.407	0.407
ZeroEntropy zerank-2	0.395	0.434	0.464	0.475	0.472

Table 19: Economics subset, Recall@20.

Reranker	k=100	k=200	k=500	k=1,000	k=2,000
Hybrid Retrieval Ceiling	0.519	0.636	0.749	0.851	0.902
Cohere rerank-v4.0-pro	0.462	0.548	0.551	0.560	0.536
Voyage rerank-2.5	0.493	0.561	0.557	0.581	0.575
ZeroEntropy zerank-2	0.481	0.550	0.576	0.621	0.606

Table 20: Psychology subset, Recall@20.

Reranker	k=100	k=200	k=500	k=1,000	k=2,000
Hybrid Retrieval Ceiling	0.423	0.564	0.708	0.762	0.829
Cohere rerank-v4.0-pro	0.338	0.382	0.410	0.395	0.379
Voyage rerank-2.5	0.359	0.403	0.443	0.433	0.414
ZeroEntropy zerank-2	0.349	0.416	0.457	0.457	0.456

Table 21: Robotics subset, Recall@20.

B.3 nDCG@10

Reranker	k=100	k=200	k=500	k=1,000	k=2,000
Cohere rerank-v4.0-pro	0.315	0.361	0.409	0.410	0.404
Voyage rerank-2.5	0.306	0.380	0.417	0.414	0.420
ZeroEntropy zerank-2	0.314	0.394	0.452	0.465	0.479

Table 22: Biology subset, nDCG@10.

Reranker	k=100	k=200	k=500	k=1,000	k=2,000
Cohere rerank-v4.0-pro	0.429	0.446	0.466	0.473	0.467
Voyage rerank-2.5	0.443	0.459	0.481	0.484	0.484
ZeroEntropy zerank-2	0.447	0.464	0.487	0.501	0.505

Table 23: Earth Science subset, nDCG@10.

Reranker	k=100	k=200	k=500	k=1,000	k=2,000
Cohere rerank-v4.0-pro	0.271	0.273	0.239	0.223	0.183
Voyage rerank-2.5	0.283	0.296	0.300	0.307	0.307
ZeroEntropy zerank-2	0.341	0.364	0.363	0.362	0.358

Table 24: Economics subset, nDCG@10.

Reranker	k=100	k=200	k=500	k=1,000	k=2,000
Cohere rerank-v4.0-pro	0.382	0.420	0.370	0.334	0.287
Voyage rerank-2.5	0.373	0.407	0.395	0.403	0.396
ZeroEntropy zerank-2	0.401	0.438	0.438	0.451	0.445

Table 25: Psychology subset, nDCG@10.

Reranker	k=100	k=200	k=500	k=1,000	k=2,000
Cohere rerank-v4.0-pro	0.251	0.256	0.247	0.236	0.224
Voyage rerank-2.5	0.256	0.279	0.301	0.301	0.289
ZeroEntropy zerank-2	0.285	0.322	0.335	0.336	0.333

Table 26: Robotics subset, nDCG@10.

B.4 Recall at Extended Output Cutoffs

	Biology	Earth Sci.	Economics	Psychology	Robotics	Mean
Hybrid Retrieval Ceiling (k=2,000)	0.907	0.830	0.886	0.902	0.829	0.871
Recall@20						
Cohere rerank-v4.0-pro	0.603	0.580	0.390	0.536	0.379	0.498
Voyage rerank-2.5	0.567	0.577	0.407	0.575	0.414	0.508
ZeroEntropy zerank-2	0.652	0.613	0.472	0.606	0.456	0.560
Recall@50						
Cohere rerank-v4.0-pro	0.716	0.654	0.504	0.675	0.458	0.601
Voyage rerank-2.5	0.695	0.675	0.544	0.695	0.498	0.621
ZeroEntropy zerank-2	0.755	0.691	0.624	0.710	0.586	0.673
Recall@100						
Cohere rerank-v4.0-pro	0.802	0.698	0.591	0.757	0.560	0.682
Voyage rerank-2.5	0.771	0.741	0.619	0.752	0.558	0.688
ZeroEntropy zerank-2	0.810	0.738	0.711	0.789	0.644	0.738

Table 27: Per-subset recall at output cutoffs of 20, 50, and 100 at first-stage depth k=2,000, underlying the mean of Table 2. The shaded row is the first-stage pool recall at k=2,000 from Table 10, which upper-bounds every cutoff. Values match the shaded Recall@100 row of Table 30.

C Singleton Listwise Reranker Results

Tables 28 and 29 report the per-subset listwise results underlying the aggregate comparison of Section 3.3, in the primary metric and Success@1 respectively.

Subset	ZeroEntropy zerank-2	GPT-5.4 Mini	GPT-5.6 Luna	GPT-5.6 Terra
Biology	0.479	0.479 ± 0.004	0.532 ± 0.006	$\mathbf{0.539 \pm 0.001}$
Earth Science	0.505	0.485 ± 0.008	0.531 ± 0.001	$\mathbf{0.553 \pm 0.006}$
Economics	0.358	0.336 ± 0.006	0.362 ± 0.005	$\mathbf{0.405 \pm 0.005}$
Psychology	0.445	0.449 ± 0.009	$\mathbf{0.524 \pm 0.008}$	0.518 ± 0.004
Robotics	0.333	0.372 ± 0.007	0.403 ± 0.003	$\mathbf{0.423 \pm 0.001}$

Table 28: Per-subset nDCG@10 after reranking the ZeroEntropy zerank-2 top-20. Unlike Recall@20, nDCG@10 is not pinned by the fixed candidate pool: reordering the retained 20 changes the top-10. Listwise values are means over three trials with across-trial standard deviations, over the same 512 shared queries.

Subset	ZeroEntropy zerank-2	GPT-5.4 Mini	GPT-5.6 Luna	GPT-5.6 Terra
Biology	0.461	0.503 ± 0.005	$\mathbf{0.569 \pm 0.014}$	0.549 ± 0.008
Earth Science	0.552	0.543 ± 0.024	0.612 ± 0.007	$\mathbf{0.652 \pm 0.004}$
Economics	0.262	0.311 ± 0.008	0.314 ± 0.012	$\mathbf{0.401 \pm 0.012}$
Psychology	0.364	0.377 ± 0.010	$\mathbf{0.492 \pm 0.025}$	0.471 ± 0.010
Robotics	0.326	0.359 ± 0.023	0.402 ± 0.009	$\mathbf{0.438 \pm 0.005}$

Table 29: Per-subset Success@1 after listwise reranking of the ZeroEntropy zerank-2 top-20. Listwise values are means over three trials with across-trial standard deviations, over the 512 queries shared by all three models. The best listwise model per subset is highlighted in bold.

D Extended Listwise Window Results

Tables 30 and 31 report the per-subset results underlying Tables 3 and 9. Input rows are the unreordered input lists and GPT-5.6 Luna rows are means across three trials with across-trial standard deviations. The shaded row in each table is the Recall@100 of the input window, which is the ceiling for every condition in that table and is invariant under reordering, since the output is a permutation of the input pool.

	Biology	Earth Sci.	Economics	Psychology	Robotics	Mean
ZeroEntropy zerank-2 Top-100 Recall	0.810	0.738	0.711	0.789	0.644	0.738
nDCG@10						
ZeroEntropy zerank-2 input ordering	0.479	0.505	0.358	0.445	0.333	0.424
Luna over top-100	0.536 ±0.011	0.539 ±0.006	0.385 ±0.004	0.534 ±0.007	0.415 ±0.006	0.482
Success@1						
ZeroEntropy zerank-2 input ordering	0.461	0.552	0.262	0.364	0.326	0.393
Luna over top-100	0.578 ±0.026	0.603 ±0.009	0.356 ±0.006	0.498 ±0.012	0.395 ±0.006	0.486
Recall@20						
ZeroEntropy zerank-2 input ordering	0.652	0.613	0.472	0.606	0.456	0.560
Luna over top-100	0.633 ±0.006	0.570 ±0.013	0.482 ±0.004	0.630 ±0.006	0.512 ±0.022	0.565

Table 30: Luna over the ZeroEntropy zerank-2 top-100 (k=2,000), per subset. The input ordering is ZeroEntropy zerank-2's retained set in descending score order. The shaded row is the Recall@100 of the retained window, which upper-bounds every condition and is unchanged by reordering. Luna cells are means across three trials with across-trial standard deviations.

	Biology	Earth Sci.	Economics	Psychology	Robotics	Mean
Hybrid Top-100 Recall	0.391	0.551	0.494	0.519	0.423	0.476
nDCG@10						
Hybrid input ordering	0.139	0.242	0.174	0.212	0.138	0.181
Luna over top-100	0.318 ±0.002	0.456 ±0.008	0.323 ±0.002	0.443 ±0.005	0.334 ±0.004	0.375
Success@1						
Hybrid input ordering	0.137	0.207	0.117	0.172	0.130	0.153
Luna over top-100	0.386 ±0.011	0.523 ±0.013	0.288 ±0.006	0.428 ±0.006	0.333 ±0.006	0.392
Recall@20						
Hybrid input ordering	0.218	0.353	0.256	0.338	0.247	0.282
Luna over top-100	0.351 ±0.011	0.489 ±0.004	0.396 ±0.009	0.491 ±0.012	0.377 ±0.005	0.421

Table 31: Luna over the hybrid top-100 with no cross-encoder, per subset. The input ordering is the hybrid fused-score order. The shaded row is the Recall@100 of the hybrid pool from Table 10, which upper-bounds every condition and is unchanged by reordering. Luna cells are means across three trials with across-trial standard deviations.

E Listwise Reranking Prompt

Our prompt follows the permutation-generation format of RankGPT [47].

System Prompt used for Listwise Reranking experiments.

```
You are RankGPT, an expert relevance ranker.  You are given a query and a numbered
list of passages.  Rank the passages from most to least relevant to the query and
return the passage numbers in descending order of relevance, as the ranking array.
Every passage number must appear exactly once.  Do not include any explanation or
per-passage reasoning -- return only the ranking.
```

User Message Template

```
Query:  {query}
Passages ({n} total):  {passages}
Rank the {n} passages above by their relevance to the query and return all {n}
passage numbers in descending order of relevance; every integer from 1 to {n} must
appear exactly once.
```

F Provider Pricing and Rate Limits

All prices and limits were retrieved from provider documentation as of July 31, 2026 and are pay-as-you-go list prices for hosted API access. Table 32 collects the billing bases and list prices, Table 33 the published rate limits and per-request caps, and Table 34 applies them to every primary condition of the paper across the depth sweep.

Table 32: Billing basis and list prices (accessed July 31, 2026).

Model	Provider	Billing basis	List price
rerank-v4.0-pro	Cohere	Per search unit: one query and up to 100 documents; a document whose length plus the query exceeds 500 tokens is split into chunks that each count as a document	$0.0025 per search unit
rerank-2.5	Voyage AI	Per token: $(q \times \#\text{docs}) + \sum$ doc tokens	$0.05 / 1M tokens
zerank-2	ZeroEntropy	Per token, with the query counted once per document scored	$0.025 / 1M tokens
GPT-5.4 Mini	OpenAI	Per token, input / output	$0.75 / $4.50 per 1M
GPT-5.6 Luna	OpenAI	Per token, input / output	$0.20 / $1.20 per 1M
GPT-5.6 Terra	OpenAI	Per token, input / output	$2.00 / $12.00 per 1M

Table 33: Published rate limits and per-request caps for the reranking providers (accessed July 31, 2026). RPM and TPM denote requests and tokens per minute. OpenAI limits for the listwise models are account-specific and are discussed in the text.

Model	**Published rate limits**	**Per-request caps**
rerank-v4.0-pro	1,000 RPM (production keys); 10 RPM (trial)	100 documents per search unit; longer documents auto-chunked
rerank-2.5	Tier 1: 2M TPM, 2,000 RPM; Tier 2: 2×; Tier 3: 3×	1,000 documents and 600K total tokens per request; 8K-token query
zerank-2	Byte-based (fast mode): Free 500K, Starter 2M, Teams 25M, Enterprise 150M UTF-8 bytes/min; 100–2,000 RPM by tier	5MB payload per request

OpenAI does not publish a single static reference for per-model rate limits. Its rate-limits documentation specifies that limits are enforced per model and per organization, advance through usage tiers on cumulative spend, and each account's exact values are surfaced in its Limits dashboard. Limits are per-model pools, so parallel calls to Mini, Luna, and Terra do not contend for a shared quota.

Table 34: Modeled per-query cost by condition and retrieval depth (d=500, q=200), at list prices as of July 31, 2026. Listwise passes are depth-independent constants, because every listwise model sees a fixed 20-document window, and are added to the cost of the condition that produces the retained top-20.

Condition	k=100	k=200	k=500	k=1,000	k=2,000
Cohere rerank-v4.0-pro	\$0.0050	\$0.0100	\$0.0250	\$0.0500	\$0.1000
Voyage rerank-2.5	\$0.0035	\$0.0070	\$0.0175	\$0.0350	\$0.0700
ZeroEntropy zerank-2	\$0.0018	\$0.0035	\$0.0088	\$0.0175	\$0.0350
CVZ fusion	\$0.0103	\$0.0205	\$0.0513	\$0.1025	\$0.2050
Listwise additions (depth-independent, applied to the retained top-20):					
+ GPT-5.4 Mini					+\$0.0083
+ GPT-5.6 Luna					+\$0.0022
+ GPT-5.6 Terra					+\$0.0220
+ Equal-weight RRF (Mini, Luna, Terra)					+\$0.0325

The formulas underlying Table 34 are as follows. At depth k with q-token queries and d-token documents:

$$\text{Voyage rerank-2.5 cost} = (q+d)\cdot k \times \$0.05/10^6,$$
$$\text{ZeroEntropy zerank-2 cost} = (q+d)\cdot k \times \$0.025/10^6,$$
$$\text{Cohere rerank-v4.0-pro cost} = \lceil c\cdot k/100\rceil \times \$0.0025,$$

where c is the number of chunks per document under Cohere's 500-token chunking rule ($c = 2$ under our assumptions, $c = 1$ if $q + d \leq 500$). Listwise per-query cost is $10{,}400$ input tokens $\times$ the input rate plus 100 output tokens $\times$ the output rate; fusion costs are the sums of their members, and the RRF/RSF aggregation itself is computationally negligible. ZeroEntropy's rate limiting, as distinct from its billing, is byte-based: bytes per query $\approx k \times (150 + |\text{query}|_8 + |\text{doc}|_8)$, where $|\cdot|_8$ denotes UTF-8 byte length. At our assumptions this is approximately 5.9MB at k=2,000, and a full-depth query must therefore be split across requests under the 5MB payload cap.