BracketRank: Large Language Model Document Ranking via Reasoning-based Competitive Elimination

Abdelrahman Abdallah, Mohammed Ali, Bhawna Piryani, Adam Jatowt
University of Innsbruck
{abdelrahman.abdallah,adam.jatowt}@uibk.ac.at

Abstract

Reasoning-intensive retrieval requires deep semantic inference beyond surface-level keyword matching, posing a challenge for current LLM-based rerankers limited by context constraints and order sensitivity. We propose **BracketRank**, a framework that treats document reranking as a reasoning-driven competitive tournament. Our approach introduces three key innovations: (1) adaptive grouping based on model context limits, (2) reasoning-enhanced prompts that mandate step-by-step relevance explanations, and (3) a bracket-style elimination structure with winner and loser tracks. This design ensures robust document advancement while enabling parallel processing across competition stages. Evaluation on the BRIGHT reasoning benchmark shows that **BracketRank** achieves **26.56 nDCG@10**, significantly outperforming state-of-the-art baselines including RankGPT-4 (17.0) and Rank-R1-14B (20.5). On TREC datasets, BracketRank achieves 77.90 nDCG@5 on DL 19 and 75.85 nDCG@5 on DL 20, exceeding all baselines, establishing that explicit reasoning within competitive elimination is a powerful paradigm for complex, multi-step retrieval tasks.[1]

1 Introduction

Real-world information retrieval often demands more than surface-level matching between queries and documents. Finding documentation for a coding problem requires understanding function logic and syntax. Answering scientific questions necessitates connecting theoretical principles to specific phenomena. These *reasoning-intensive* retrieval scenarios pose fundamental challenges that standard retrieval benchmarks fail to capture (Su et al., 2024; Abdallah et al., 2026a,b). Existing benchmarks primarily consist of information-seeking queries where keyword or semantic matching suffices (Bajaj et al., 2016; Chen et al., 2017; Thorne et al., 2018; Lewis et al., 2020). On the other hand, the BRIGHT benchmark (Su et al., 2024) reveals that even state-of-the-art retrieval models achieve only 18.0 nDCG@10 on queries requiring intensive reasoning, which is dramatically lower than their performance on conventional benchmarks (Bajaj et al., 2016; Thakur et al., 2021).

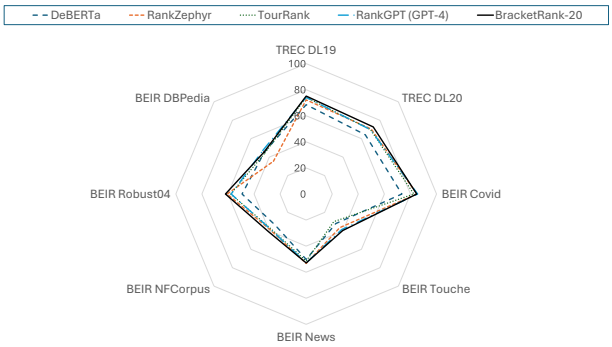


Figure 1: Radar chart comparing nDCG@5 performance of top reranking methods, including DeBERTa, RankZephyr, RankGPT (GPT-4), and **BracketRank**-20 (GPT-4), across TREC DL20, TREC DL19 and BEIR datasets (Covid, NFCorpus, Touche, DBPedia, News, Robust04).

Large Language Models (LLMs) offer a promising direction for reasoning-intensive retrieval through their demonstrated capabilities in complex reasoning tasks (Wei et al., 2022). LLM-based document ranking methods including pointwise (Sachan et al., 2022; Zhuang et al., 2023a), pairwise (Qin et al., 2023), and listwise (Sun et al., 2023; Ma et al., 2023) approaches have shown strong zero-shot performance on traditional benchmarks. However, these methods face three key limitations that become especially problematic for reasoning-intensive tasks. First, context length constraints prevent processing many documents simultaneously, limiting the scope of comparative reasoning. Second, sequential decoding

[1] https://github.com/DataScienceUIBK/BracketRank

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 3381–3397

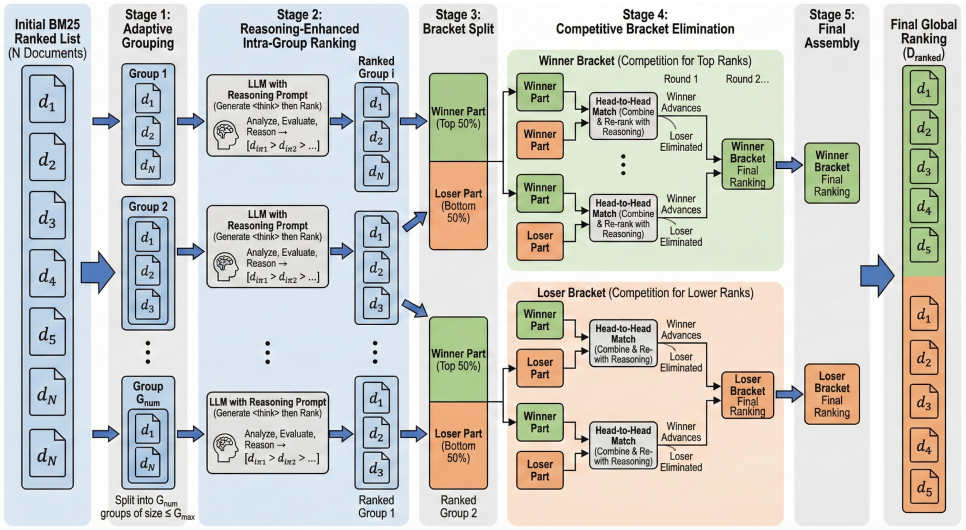


Figure 2: Overview of the **BracketRank** framework. The process consists of five stages: (1) adaptive grouping of initial retrievals; (2) intra-group ranking using LLMs with explicit reasoning prompts; (3) splitting ranked groups into winner and loser tracks; (4) parallel competitive bracket elimination where winners advance via head-to-head matches; and (5) assembly of the final global ranking.

within listwise prompts creates bottlenecks that prevent parallelization. Third, ranking results depend heavily on initial document order, undermining the consistency needed for complex reasoning.

We propose **BracketRank**, a reasoning-driven competitive elimination framework specifically designed for reasoning-intensive retrieval. Our key insight is that complex queries benefit from explicit, structured reasoning during document comparison—not just in final ranking decisions, but throughout a systematic competition process. **BracketRank** combines three innovations: (1) adaptive grouping that optimizes group sizes based on LLM context limits, (2) reasoning-enhanced prompts that require step-by-step relevance explanations for each comparison, and (3) bracket-style elimination where documents compete through winner and loser brackets with explicit reasoning at each stage.

The competitive bracket structure provides two critical advantages for reasoning-intensive tasks. First, it enforces multiple rounds of deliberate comparison, allowing the LLM to reason about document relevance from different perspectives as documents face various opponents. Second, the winner/loser bracket design ensures documents receive fair evaluation regardless of initial positioning—particularly important when first-stage retrievers fail to identify relevant documents for complex queries.As shown in Figure 1, **BracketRank** achieves superior performance across diverse datasets, with particularly strong gains on reasoning-intensive benchmarks.

Our contributions are as follows: **(1)** We introduce **BracketRank**, a competitive elimination framework that combines explicit reasoning requirements with systematic bracket competition, specifically designed for reasoning-intensive retrieval tasks. **(2)** We design reasoning-augmented prompts that improve ranking consistency by requiring step-by-step relevance explanations, enabling LLMs to perform deliberate comparative reasoning during document evaluation. **(3)** Extensive experiments demonstrate that **BracketRank** achieves state-of-the-art performance on the BRIGHT benchmark for reasoning-intensive retrieval while maintaining competitive results on traditional benchmarks, establishing explicit reasoning as a key mechanism for complex retrieval tasks.

2 Related Work

Document ranking has evolved from BERT-based cross-encoders (Abdallah et al., 2025a; Nogueira and Cho, 2019; Devlin et al., 2018) to T5-based sequence models (Nogueira et al., 2020; Raffel et al., 2020). LLM-based approaches fall into three categories: *pointwise* (Sachan et al., 2022; Liang et al., 2022), *pairwise* (Mozafari and Jatowt, 2025; Qin et al., 2023; Pradeep et al., 2021), and *listwise* (Mozafari and Jatowt, 2025; Sun et al., 2023; Ma et al., 2023). Listwise methods like

RankGPT (Sun et al., 2023) avoid quadratic pairwise costs but suffer from order sensitivity and sequential processing constraints. Tournament-inspired methods include ListT5 (Yoon et al., 2024) (requires MS MARCO training) and TourRank (Chen et al., 2025) (multi-round points accumulation). Our **BracketRank** differs by using single-elimination brackets with explicit reasoning requirements for stable decisions. Recent work on reasoning for ranking (Ji et al., 2024) focuses on explanation generation rather than consistency. See § B for detailed discussion.

3 Method

In this section, we introduce **BracketRank**, a reasoning-driven competitive elimination approach for zero-shot document ranking. Our method addresses the core limitations of existing listwise approaches through adaptive group formation and systematic bracket competition. We split ranked groups into winner and loser tracks and perform head-to-head matches by *combining two groups and re-ranking their documents* with the same listwise prompt. Winners advance; losers are eliminated (single-elimination), yielding a global order after $\lceil \log_2 G \rceil$ rounds. **BracketRank** combines reasoning-enhanced prompts with tournament-style elimination to achieve robust rankings while maintaining efficiency. Figure 2 illustrates the complete methodology. Figure 3 illustrates the complete methodology. We first explain how we form adaptive groups based on LLM context constraints. Then we describe the reasoning-enhanced ranking process within each group. Next, we detail the competitive bracket elimination that creates head-to-head competition between groups. Finally, we present the complete algorithm that integrates these components.

We first explain how we form adaptive groups based on LLM context constraints. Then we describe the reasoning-enhanced ranking process within each group. Next, we detail the competitive bracket elimination that creates head-to-head competition between groups. Finally, we present the complete algorithm that integrates these components.

3.1 Adaptive Group Formation

Given N candidate documents returned by the first-stage retriever, we determine the number and sizes of groups from the LLM's context budget.

Algorithm 1 BracketRank Algorithm: Adaptive group formation, reasoning-enhanced intra-group ranking, bracket split into winner and loser tracks, iterative elimination competition, and final ranking assembly

Input: Query q, candidate documents $D = \{d_1, \ldots, d_N\}$, max group size G_{max}
Output: Ranked document list D_{ranked}
// Adaptive group formation
$G_{num} \leftarrow \lceil N/G_{max} \rceil$
$Groups \leftarrow$ SplitIntoGroups(D, G_{num})
// Reasoning-enhanced group ranking
$RankedGroups \leftarrow []$
for each G_i in $Groups$ **do**
  $R_i \leftarrow$ ReasoningRank(q, G_i)
  $RankedGroups$.append(R_i)
end for
// Create initial winner and loser brackets
$WinnerBracket \leftarrow []$, $LoserBracket \leftarrow []$
for each R_i in $RankedGroups$ **do**
  $W_i, L_i \leftarrow$ SplitGroup(R_i) // Top half vs bottom half
  $WinnerBracket$.append(W_i)
  $LoserBracket$.append(L_i)
end for
// Iterative bracket elimination
$WinnerResult \leftarrow$ RunIterativeBracket$(WinnerBracket, q)$
$LoserResult \leftarrow$ RunIterativeBracket$(LoserBracket, q)$
// Combine final ranking
$D_{ranked} \leftarrow WinnerResult + LoserResult$
return D_{ranked}

Function RunIterativeBracket$(Bracket, q)$:
while $|Bracket| > 1$ **do**
  $NextRound \leftarrow []$
  for $i = 0$ to $|Bracket| - 1$ step 2 **do**
    if $i + 1 < |Bracket|$ **then**
      $Combined \leftarrow Bracket[i] + Bracket[i+1]$
      $Ranked \leftarrow$ ReasoningRank$(q, Combined)$
      $Winner, Loser \leftarrow$ SplitGroup$(Ranked)$
      $NextRound$.append$(Winner)$ // Only winners advance
    else
      $NextRound$.append$(Bracket[i])$ // Odd group advances
    end if
  end for
  $Bracket \leftarrow NextRound$
end while
return $Bracket[0]$ // Final ranked list

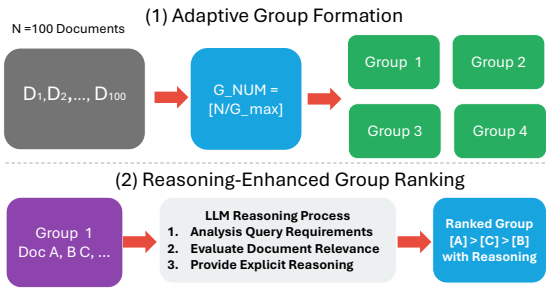
Let $G_{\max}$ denote the maximum number of documents that can fit into a single listwise prompt (respecting the model's token limit). We set the number of groups to

$$G_{\text{num}} = \left\lceil \frac{N}{G_{\max}} \right\rceil. \tag{1}$$

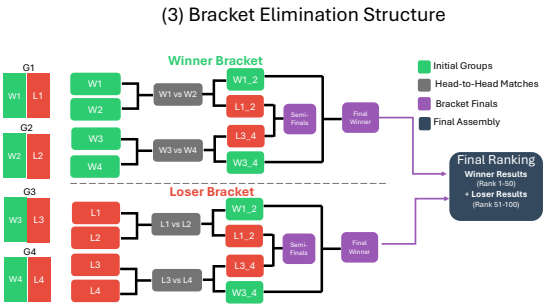
We partition the retriever-ranked list into G_{num} *contiguous* slices while keeping sizes as even as possible and never exceeding $G_{\max}$. Let

$$s = \left\lfloor \frac{N}{G_{\text{num}}} \right\rfloor, \qquad r = N \bmod G_{\text{num}}.$$

We create G_{num} groups in order: the first r groups receive $s+1$ documents each and the remaining



(a) Adaptive group formation and reasoning-enhanced ranking within groups



(b) Competitive bracket elimination structure with winner and loser brackets

Figure 3: Detailed breakdown of **BracketRank** methodology components. (a) illustrates the adaptive grouping based on context constraints. (b) details the specific document flow through the competitive elimination tracks.

$G_{\text{num}}-r$ groups receive s documents each. Denoting the size of group i by m_i, we have

$$m_i = \begin{cases} s+1, & 1 \le i \le r, \\ s, & r+1 \le i \le G_{\text{num}}. \end{cases}$$

This construction preserves the retriever order within every group and ensures that group-size differences are at most one. Because $G_{\text{num}} = \lceil N/G_{\max} \rceil$, it follows that $s+1 \le G_{\max}$, so no group exceeds the per-prompt limit. Concretely, if the groups are contiguous in the ranked list, their index ranges satisfy

$$a_1 = 1, \quad b_i = a_i + m_i - 1,$$
$$a_{i+1} = b_i + 1 \quad (i = 1, \ldots, G_{\text{num}}-1).$$

Given an LLM token budget B for a single call, an average passage length $\bar{L}$ (in tokens), prompt overhead T (instructions, delimiters), and per-document framing cost H (e.g., identifiers and separators), a conservative choice is

$$G_{\max} = \left\lfloor \frac{B - T}{\bar{L} + H} \right\rfloor.$$

In our experiments we set $G_{\max}$=20 by estimating $(\bar{L}, T, H)$ on the development split; **BracketRank** is not sensitive to this choice within a reasonable range (see the group-size study in Fig. 5).

3.2 Reasoning-Enhanced Group Ranking

Each group undergoes independent ranking using reasoning-augmented prompts. This step transforms a relevance assessment into explicit reasoning about document quality relative to the query.

For a query q and group $G_i = \{d_{i1}, d_{i2}, \ldots, d_{ik}\}$, the reasoning prompt structure is shown in Figure 8. This reasoning component serves two purposes. First, it improves ranking by forcing the LLM to articulate its decision process. Second, it provides transparency into why certain documents rank higher than others.

Each group produces a ranked list $R_i = [d_{i\pi_1}, d_{i\pi_2}, \ldots, d_{i\pi_k}]$ where π represents the ranking permutation. These group rankings form the input to the competitive elimination stage.

Stage	Number of Docs	Ranking Position
Initial Groups	$G \times D_g = N$	Initial BM25 order
Winner Groups	$G \times \lceil D_g/2 \rceil$	Ranks 1 to $\sum W_i$
Loser Groups	$G \times \lfloor D_g/2 \rfloor$	Ranks $\sum W_i + 1$ to N
Winner Champion	W_{final}	Rank 1
Winner Runner-up	W_{final}	Ranks 2 to $2 \times W_{final}$
$\vdots$	$\vdots$	$\vdots$
Loser Champion	L_{final}	Rank $\sum W_i + 1$
Loser Final	L_{final}	Ranks $N - L_{final} + 1$ to N

Table 1: Document distribution and ranking structure in BracketRank. Groups are split into winner and loser brackets, with final rankings determined by bracket competition.

3.3 Competitive Bracket Elimination

The core innovation of **BracketRank** lies in its competitive bracket structure. Rather than simply combining group results, we create systematic head-to-head competition between groups through winner and loser brackets.

Winner and Loser Bracket Formation. Each ranked group R_i is split at the midpoint. The top half enters the winner bracket W_i while the bottom half enters the loser bracket L_i. This creates two parallel competition tracks:

$$W_i = \{d_{i\pi_1}, d_{i\pi_2}, \ldots, d_{i\pi_{k/2}}\} \quad (2)$$
$$L_i = \{d_{i\pi_{(k/2)+1}}, d_{i\pi_{(k/2)+2}}, \ldots, d_{i\pi_k}\} \quad (3)$$

Head-to-Head Competition. Groups compete in pairs through direct ranking. When two winner groups W_i and W_j compete, their documents are combined and re-ranked using the same reasoning-enhanced prompt. The winning group advances while the losing group is eliminated. This pairwise competition continues until each bracket produces a final ranking.

Final Ranking Assembly. The complete ranking combines results from both brackets. Winner bracket results form the top portion of the final ranking, followed by loser bracket results. This guarantees that documents competing in the winner bracket receive higher positions than those in the loser bracket. Table 1 shows how documents flow through the BracketRank structure. The adaptive grouping ensures balanced competition while the bracket elimination creates systematic advancement paths for all documents.

3.4 The Complete BracketRank Algorithm

Algorithm 1 operates in four phases. First, documents are divided into $G_{num} = \lceil N/G_{max} \rceil$ groups using contiguous splitting (lines 3-4). Second, each group undergoes independent reasoning-enhanced ranking that can be fully parallelized (lines 6-9). Third, ranked groups split at the midpoint into winner and loser brackets (lines 11-16). Fourth, iterative bracket elimination runs head-to-head competitions where groups combine and re-rank, with winners advancing until one group remains per bracket (lines 18-19). The algorithm supports two-level parallelisation. Initial group ranking processes all groups simultaneously. Bracket elimination processes all matches at the same round level in parallel. The computational complexity is $O(\log G_{num})$ for bracket elimination, making it significantly more efficient than $O(N^2)$ pairwise methods. Final ranking concatenates winner bracket results followed by loser bracket results, ensuring systematic relevance-based positioning. The competitive bracket structure (detailed in Algorithm 1) ensures systematic advancement. We explored alternative tournament formats including double-elimination and round-robin (see § 5.6), but single-elimination provides optimal performance-efficiency trade-offs.

4 Experiments

4.1 Experimental Settings

Datasets We evaluate BracketRank on benchmarks spanning both traditional and reasoning-intensive retrieval.

Reasoning-Intensive Retrieval. BRIGHT (Su et al., 2024) is the first benchmark specifically designed for reasoning-intensive retrieval, containing 1,384 real-world queries across 12 diverse datasets: Biology, Earth Science, Economics, Psychology, Robotics, StackOverflow, Sustainable Living (from StackExchange), LeetCode, Pony (coding domains), AoPS (math competitions), and TheoremQA-Theorem/Question.

Traditional Benchmarks. TREC DL 19 and DL 20 (Craswell et al., 2020, 2021) contain 43 and 54 queries respectively for passage ranking evaluation. **BEIR** (Thakur et al., 2021) provides heterogeneous zero-shot evaluation across eight diverse domains: Covid, NFCorpus, Touche, DBPedia, SciFact, Signal, News, and Robust04. **NovelEval-2306** (Sun et al., 2023) provides 21 queries from domains published after GPT-4's training cutoff, ensuring true zero-shot evaluation.

Metrics We re-rank the top-100 documents retrieved by BM25 using PySerini and Rankify implementation (Lin et al., 2021; Abdallah et al., 2025b). Our evaluation uses NDCG@1, 5, 10 as primary metrics, following standard practice in document ranking evaluation.

Baselines We compare against supervised and zero-shot rerankers. **Supervised:** monoBERT (Nogueira and Cho, 2019), monoT5 (220M/3B) (Nogueira et al., 2020), and Cohere Rerank-v2. **Zero-shot LLM:** B-RG (Liang et al., 2022), PRP-Allpair (Qin et al., 2023), Setwise (Zhuang et al., 2023c), RankGPT (Sun et al., 2023), TourRank (Chen et al., 2025), and Rank-R1 (GRPO-trained) (Zhuang et al., 2025).

4.2 Main Results

4.2.1 Performance on Reasoning-Intensive Retrieval

Table 2 presents results on the BRIGHT benchmark for reasoning-intensive retrieval. **BracketRank** achieves state-of-the-art performance with 26.6 average nDCG@10, substantially outperforming all baselines including the previous best Rank-R1-14B with GRPO training (20.5) and RankGPT-4 (17.0). **BracketRank**

Method	StackExchange							Coding		Theorem-based			Avg
	Bio.	Earth.	Econ.	Psy.	Rob.	Stack.	Sus.	Leet.	Pony	AoPS	TheoQ.	TheoT.	
BM25	18.2	27.9	16.4	13.4	10.9	16.3	16.1	24.7	4.3	6.5	7.3	2.1	13.7
Distilled / Fine-tuned Rerankers													
RankZephyr-7B	21.9	23.7	14.4	10.3	7.6	13.7	16.6	24.7	6.5	6.8	7.3	2.0	13.0
Setwise-3B (SFT)	22.0	18.8	10.4	11.5	9.1	5.8	16.7	9.9	5.7	4.0	3.8	3.4	10.1
Setwise-7B (SFT)	28.7	30.1	14.1	23.9	18.9	13.7	19.6	20.7	7.1	7.0	8.2	8.2	16.7
Setwise-14B (SFT)	22.0	29.3	15.4	23.0	20.1	15.7	20.3	19.4	6.2	9.5	9.7	9.9	16.7
Rank-R1-3B (GRPO)	18.4	17.1	13.7	16.9	9.0	10.0	16.5	11.1	4.7	3.5	3.2	5.9	10.8
Rank-R1-7B (GRPO)	26.0	28.5	17.2	24.2	19.1	10.4	24.2	19.8	4.3	4.3	8.3	10.9	16.4
Rank-R1-14B (GRPO)	31.2	38.5	21.2	26.4	22.6	18.9	27.5	20.2	9.2	9.7	9.2	11.9	20.5
Zero-Shot Rerankers													
Setwise-3B	14.3	17.5	12.0	10.2	7.7	7.9	15.4	15.4	5.3	1.7	2.1	4.2	9.5
Setwise-7B	23.6	22.3	16.1	17.1	14.9	9.2	18.3	14.9	6.3	4.1	5.6	10.4	13.6
Setwise-14B	29.5	32.2	20.5	24.8	18.9	14.7	23.6	18.7	8.7	8.0	7.6	9.3	18.0
Rank-R1-3B	13.7	17.3	11.9	15.2	10.0	6.6	17.8	7.7	3.7	4.0	2.5	6.0	9.7
Rank-R1-7B	26.8	24.8	17.9	22.1	17.4	10.3	21.1	15.6	4.4	3.3	5.9	10.4	15.0
Rank-R1-14B	30.1	36.6	22.1	24.6	21.7	15.4	25.0	17.0	9.0	9.1	9.2	11.6	19.3
RankGPT-4†	33.8	34.2	16.7	27.0	22.3	27.7	11.1	3.4	15.6	1.2	0.2	8.6	17.0
BracketRank-20 (GPT-4)	**38.2****	**45.2*****	**22.7***	**36.5*****	**30.5*****	**26.2****	**33.8*****	**16.9*****	**28.7****	**9.2**	**14.2****	**16.6*****	**26.6*****

Table 2: BRIGHT nDCG@10 results for reasoning-intensive retrieval. All methods rerank BM25 top-100 documents. Statistical significance vs. best baseline: *p<0.05, **p<0.01, ***p<0.001. ^Results from original paper using different BM25 system. **BracketRank** achieves substantial improvements across all domains, with particularly strong gains on scientific reasoning (Biology, Earth Science, Psychology) and coding tasks (LeetCode, StackOverflow).

Methods	Model	TREC DL 19		TREC DL 20	
		NDCG@5	NDCG@10	NDCG@5	NDCG@10
BM25	-	52.78	50.58	50.67	47.96
Supervised Methods					
monoBERT	BERT (340M)	73.25	70.50	70.74	67.28
monoT5 (220M)	T5 (220M)	**73.77**	71.48	69.40	66.99
monoT5	T5 (3B)	73.74	**71.83**	**72.32**	**68.89**
Zero-Shot LLM Methods					
B-RG	GPT3.5	63.33	62.51	65.04	63.37
PRP-Allpair	GPT3.5	70.43	68.18	69.75	66.40
Setwise.heapsort (c=10)	GPT3.5	70.55	68.16	57.05	53.73
Setwise.bubblesort (c=10)	GPT3.5	67.62	66.19	57.03	53.82
RankGPT	GPT3.5	72.05	68.19	67.25	63.60
RankGPT	GPT-4	75.98**	72.67**	72.32	69.48*
TourRank	GPT3.5	**73.83***	**71.63***	**72.49***	**69.56***
TourRank	GPT-4	75.57**	74.13***	72.46**	69.79**
BracketRank −10	GPT-4	**79.15*****	65.66	**75.29*****	64.04
BracketRank −15	GPT-4	**77.53*****	71.65	**75.25*****	67.56
BracketRank −20	GPT-4	**77.90*****	**75.11*****	**75.85*****	**72.75*****

Table 3: Performance comparison on TREC datasets. Best supervised and zero-shot methods shown in bold. Statistical significance vs. best baseline: *p<0.05, **p<0.01, ***p<0.001.

Method	Covid	NFCorpus	Touche	DBPedia	SciFact	Signal	News	Robust04	BEIR (Avg)
BM25	59.47	30.75	**44.22**	31.80	67.89	33.05	39.52	40.70	43.42
Supervised									
monoBERT (340M)	70.01	36.88	31.75	41.87	71.36	31.44	44.62	49.35	47.16
monoT5 (220M)	78.34	37.38	30.82	42.42	73.40	31.67	46.83	51.72	49.07
monoT5 (3B)	80.71	38.97	32.41	44.45	76.57	32.55	48.49	56.71	51.36
Cohere Rerank-v2	81.81	36.36	32.51	42.51	74.44	29.60	47.59	50.78	49.45
ListT5 (3B)	84.70	37.70	33.60	46.20	77.00	33.80	53.20	57.80	53.00
Zero-Shot									
UPR (3B)	68.11	35.04	19.69	30.91	72.69	31.91	43.11	42.43	42.99
Promptagator++	76.2	37.0	38.1	43.4	73.1	-	-	-	-
RankGPT-3.5	76.67	35.62	36.18	44.47	70.43	32.12	48.85	50.62	49.37
RankGPT-4^	85.51	38.47	38.57	47.12	74.95	34.40	52.89	57.55	53.68
RankGPT (GPT-4)	84.92	39.05	36.47	45.79	77.61	34.20	51.24	60.98	53.78
TourRank (GPT-4)	82.59	37.99	29.98	44.64	72.17	30.83	51.46	57.87	50.94
BracketRank (GPT-4) −15	80.46	40.04*	35.94*	43.95	77.91*	32.91	49.85	59.57*	52.58*
BracketRank (GPT-4) −20	84.82	**40.86****	**39.67****	45.58	**78.03****	33.49	52.88*	**61.94****	**54.66****

Table 4: nDCG@10 performance on BEIR datasets (CovidQA, NFCorpus, Touche, DBPedia, SciFact, Signal, News, Robust04). Statistical significance vs. best baseline: *p<0.05, **p<0.01, ***p<0.001. RankGPT^: GPT-3.5 pre-rerank → GPT-4 on top-30 (original pipeline). RankGPT (GPT-4): direct rerank on top-100 BM25 for fairness.

shows improvements across all 12 reasoning-intensive domains. The gains are particularly pronounced on scientific reasoning tasks: Biology (+7.0 points over Rank-R1-14B), Earth Science (+6.7 points), and Psychology (+10.1 points). On coding-related datasets, **BracketRank** achieves 28.7 nDCG@10 on LeetCode (+8.5 points over RankGPT-4's cascaded approach) and 26.2 on StackOverflow. Understanding code requires analyzing function logic and syntax relationships precisely the type of multi-step reasoning that **BracketRank**'s explicit reasoning prompts facilitate. Our Method provides substantial improvements on domains requiring multi-step inference, such as Biology and Psychology. A more detailed analysis of how explicit reasoning benefits specific BRIGHT categories is provided in Appendix E.

4.2.2 Performance on Traditional and Novel Benchmarks

BracketRank achieves state-of-the-art results across traditional and novel benchmarks, outperforming supervised and zero-shot baselines. On **TREC DL 19 and DL 20**, **BracketRank**-20 using GPT-4 reaches **77.90** and **75.85 nDCG@5**, respectively, significantly exceeding **RankGPT-4** (75.98/72.32) and **TourRank** (75.57) with statistical significance ($p < 0.001$). Notably, it surpasses supervised models like **monoT5 (3B)** in a purely zero-shot setting. On **BEIR**,

BracketRank-20 attains a leading average of **54.66 nDCG@10**, outperforming **RankGPT-4** (53.68) and **monoT5 (3B)** (51.36). It shows exceptional robustness in specialized domains, securing top scores in **NFCorpus** (40.86), **Touche** (39.67), **SciFact** (78.03), and **Robust04** (61.94). Furthermore, on **NovelEval-2306**, **BracketRank** achieves an average **88.76 nDCG**, outperforming **RankGPT-4** (87.88) and confirming its effectiveness on content published after the model's training cutoff.

Method	prev.	Top-K	nDCG@1	nDCG@5	nDCG@10	Avg
BM25	-	-	33.33	45.96	55.77	45.02
monoBERT (340M)	BM25	100	78.57	70.65	77.27	75.50
monoT5 (220M)	BM25	100	83.33	77.46	81.27	80.69
monoT5 (3B)	BM25	100	83.33	78.38	84.62	82.11
RankGPT-3.5	BM25	100	76.19	74.15	75.71	75.35
RankGPT-4	RankGPT-3.5	20	85.71	87.49	90.45	87.88
BracketRank (GPT-4)	BM25	100	**86.42**	**89.39**	**90.47**	**88.76**

Table 5: Reranking results on NovelEval-2306. We compare BM25, monoT5, GPT baselines, and **BracketRank**.

Table 5 shows results on NovelEval-2306 (Sun et al., 2023), containing queries from domains published after GPT-4's training cutoff to ensure true zero-shot evaluation. **BracketRank** achieves state-of-the-art performance with 88.76 average NDCG versus RankGPT-4's 87.88. Strong performance at NDCG@1 (86.42) and NDCG@5 (89.39) indicates the bracket competition structure effectively identifies relevant documents even without prior exposure to the content domain.

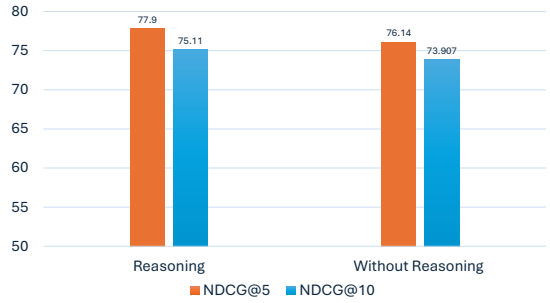


Figure 4: Ablation study showing the impact of reasoning-enhanced prompts on BracketRank performance. Reasoning requirements provide consistent improvements across both NDCG@5 and NDCG@10 metrics on TREC DL 19. In § A.2, we provide the reasoning versus bracket structure contributions on BEIR.

Method	Model	@1	@5	@10	Avg
BM25	-	54.26	52.78	50.58	52.54
BracketRank	GPT-4	79.07	77.90	75.11	77.36
BracketRank	Qwen2.5-7B	76.36	72.32	67.58	72.09
BracketRank	Llama-3.1-8B	69.77	66.42	63.84	66.68
BracketRank	Gemini-2.5-flash	78.68	74.88	71.90	75.15
BracketRank	Gemini-2.5-pro	79.46	74.41	72.86	75.58

Table 6: Results of NDGC for different LLMs on DL-19.

5 Additional Analysis

5.1 Impact of Reasoning Component

To validate the effectiveness of reasoning-enhanced prompts, we conducted an ablation study on TREC DL 19 comparing BracketRank with and without explicit reasoning requirements. Figure 4 shows consistent improvements from the reasoning component. NDCG@5 increases from 76.14 to 77.90 (a gain of 1.76 points), while NDCG@10 improves from 73.91 to 75.11 (a gain of 1.20 points). The reasoning requirements force LLMs to articulate their relevance judgments explicitly, reducing inconsistencies that occur with implicit decisions. This structured approach helps models consider multiple relevance signals systematically and creates more stable rankings when documents encounter. The reasoning requirements force LLMs to articulate their relevance judgments explicitly, reducing inconsistencies that occur with implicit decisions. This structured approach helps models consider multiple relevance signals systematically and creates more stable rankings. We provide additional analysis of reasoning versus bracket structure contributions on BEIR datasets in § A.2.

Model	**Method**	**NDCG@5**	**NDCG@10**
Qwen2.5-7B	TourRank	67.24	64.48
	BracketRank	**72.32** (+5.08)	**67.58** (+3.10)
Llama-3.1-8B	TourRank	65.09	62.35
	BracketRank	**66.42** (+1.33)	**63.84** (+1.49)

Table 7: Cross-LLM comparison on TREC DL19 showing **BracketRank** consistently outperforms TourRank across different model backends.

5.2 Cross-LLM Robustness Analysis

We evaluate **BracketRank** across proprietary and open-source LLMs to assess framework generalizability. Table 6 shows performance ranges from 66.68 (Llama-3.1-8B) to 77.36 (GPT-4) on TREC DL-19, demonstrating that the competitive elimination framework leverages fundamental

reasoning abilities rather than model-specific features. Even the open-source Qwen2.5-7B achieves 72.09, making **BracketRank** viable for resource-constrained deployments. Direct comparison with TourRank (Table 7) confirms advantages persist across backends: +5.08 NDCG@5 with Qwen2.5-7B and +1.33 with Llama-3.1-8B. More capable LLMs better exploit the bracket structure, but smaller models still benefit from systematic elimination.

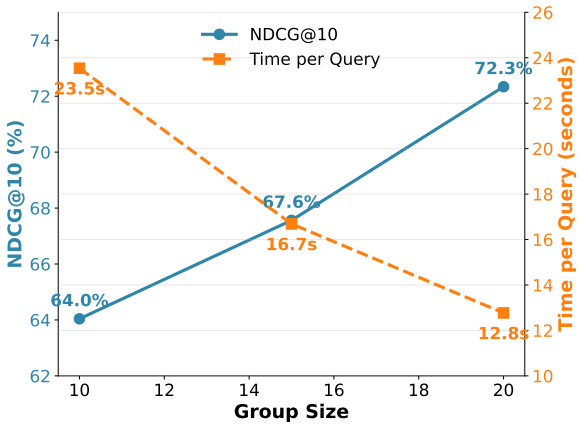


Figure 5: Group Size Impact on Per-Query Performance and Efficiency

5.3 Group Size and Complexity

Larger group sizes improve both performance and efficiency (Figure 5). BracketRank-20 achieves 72.34% NDCG@10 versus 64.04% for BracketRank-10, while reducing processing time from 23.54 to 12.77 seconds per query (45.7% reduction). Larger groups enable more comprehensive intra-group comparisons and fewer bracket elimination rounds. **BracketRank** requires $O(\log G_{num})$ rounds with approximately $N(1+\log G_{num})$ document inputs. For $N = 100$ and $G_{max} = 20$, this yields 332 inputs—more efficient than pairwise methods while maintaining superior quality. Detailed complexity analysis is in § A.1.

5.4 Retriever Robustness

While our main setup uses BM25, we also test with the dense retriever Contriever (Izacard et al., 2021). Table 8 shows that **BracketRank** retains its edge under dense retrieval: it averages 73.48 NDCG@10 on TREC DL, vs. 71.98 for TourRank-10 and 69.09 for RankGPT. The +4.39 vs. RankGPT and +1.50 vs. TourRank gains indicate the tournament design is not tied to a sparse retriever.

Method	Retriever	DL 19 NDCG@10	DL 20 NDCG@10	Avg
Contriever	-	62.02	63.42	62.72
RankGPT	Contriever	69.70	68.47	69.09
TourRank-2	Contriever	69.12	71.89	70.51
TourRank-10	Contriever	70.77	73.19	71.98
BracketRank-20	Contriever	**72.32**	**74.64**	**73.48**

Table 8: Performance with dense retriever (Contriever) on TREC datasets, demonstrating robustness across different retrieval paradigms.

Approach	NDCG@5	NDCG@10	Description
Listwise Only	71.23	68.45	Groups ranked, then concatenated
Pairwise Groups	67.12	64.88	Pairwise group comparisons
BracketRank (Full)	**75.85**	**72.75**	Reasoning + bracket elimination

Table 9: Component ablation on TREC DL20 isolating contributions of bracket elimination structure vs. reasoning-enhanced group ranking.

5.5 Ablation and Efficiency Analysis

Ablations on DL 20 (Table 9) confirm that combining intra-group listwise ranking and bracketed elimination is essential, yielding **+4.62** and **+8.73 nDCG@5** gains over group concatenation and pairwise variants, respectively. Single-elimination provides the optimal performance-efficiency trade-off compared to double-elimination or round-robin formats. Regarding computational costs (Table 12), **BracketRank**-20 requires **13 API calls** (360 total documents), providing greater comparison depth than RankGPT (9 calls, 200 documents) while using **7.7× fewer calls** than TourRank-10 to achieve superior quality. See § A.3 for detailed analysis.

5.6 Bracket Structure Comparison

We evaluate three different competitive structures to validate our single elimination design choice. The structures are defined as follows:

Single Elimination: Our current approach where groups compete once, with winners and losers split into separate brackets that run parallel elimination tournaments. **Double Elimination**: Groups receive two chances before elimination. Losers from the winner bracket drop to a loser bracket, and only groups that lose twice are eliminated completely. **Round Robin**: Every group competes against every other group in exhaustive pairwise comparisons. Final rankings are determined by win-loss records and average document positions across all matches.

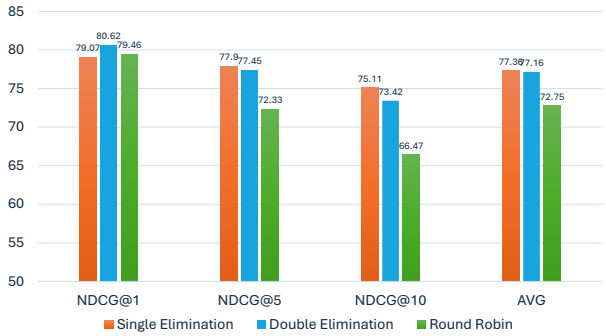


Figure 6: Comparison of bracket elimination structures on TREC DL 19. Single elimination achieves the best overall performance across all NDCG

Figure 6 compares performance across these bracket strategies on TREC DL 19. Single elimination achieves the best overall performance (77.36 average NDCG), demonstrating optimal balance across all ranking cutoffs. Double elimination reaches the highest precision at NDCG@1 (80.62), suggesting that second chances help identify the most relevant documents. However, performance drops significantly at higher cutoffs (NDCG@5: 77.45, NDCG@10: 73.42), indicating that additional complexity introduces ranking inconsistencies. Round robin shows substantial performance degradation, particularly at NDCG@10 (66.47). While exhaustive comparison seems theoretically superior, the excessive pairwise competition creates noise that degrades ranking quality. The $O(n^2)$ computational overhead also makes this approach impractical for larger document sets.

6 Efficiency-Effectiveness Trade-off

To evaluate the practical utility of **BracketRank**, we analyze the trade-off between ranking effectiveness (NDCG@10) and computational overhead. Figure 7 illustrates this balance across two dimensions: **API Efficiency** (number of requests) and **Computational Cost** (total documents processed).

As shown in Figure 7(a), **BracketRank**-20 effectively **breaks the existing Pareto front** defined by current state-of-the-art methods. While RankGPT is efficient, its effectiveness is limited by its sliding-window approach. Conversely, TourRank-10 improves effectiveness at an $11\times$ higher API cost. **BracketRank**-20 achieves superior performance (72.75% NDCG@10) while requiring only 13 API calls—an 87% reduction

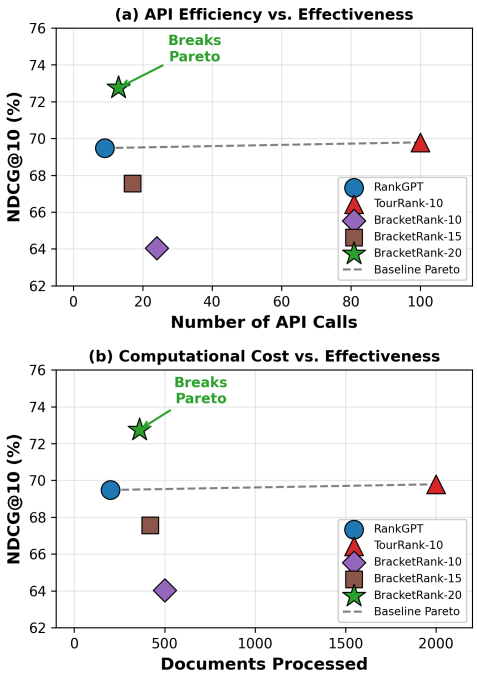


Figure 7: Efficiency-Effectiveness Pareto analysis on TREC DL20.

in cost compared to TourRank-10. This confirms that our reasoning-driven competitive elimination provides a superior "exchange rate" of 0.82 NDCG points per additional API call relative to RankGPT. For a detailed theoretical derivation and exchange rate analysis, see **Appendix D**.

7 Conclusion

We introduced **BracketRank**, a reasoning-driven competitive elimination framework for LLM-based document ranking. Our method combines adaptive grouping, reasoning-enhanced prompts, and systematic bracket competition to address key limitations in existing listwise approaches. **BracketRank** achieves state-of-the-art performance with 77.90 NDCG@5 on TREC DL 19 and 54.66 average NDCG@10 across BEIR datasets.

Acknowledgments

The authors would like to acknowledge the financial support provided by the Austrian Research Agency (FFG) for the project "AI Enabled Sustainability Jurisdiction Demonstrator" (project No. 915229).

The computational results presented here have been achieved (in part) using the LEO HPC infrastructure of the University of Innsbruck.

Limitations

While **BracketRank** achieves state-of-the-art performance across reasoning-intensive benchmarks, its deployment involves standard trade-offs common to advanced LLM-based reranking frameworks.

First, the method is subject to **inherent model constraints** such as context window limits and API latency. While our adaptive grouping strategy ensures compatibility with diverse model architectures, extremely long document collections may require standard truncation or multi-stage processing. These constraints are characteristic of all LLM-based listwise approaches and are typically mitigated by the continuous advancement of underlying model capacities and hardware acceleration.

Second, the **reasoning-enhanced prompts** are optimized for current instruction-following LLMs. While we observed consistent gains across various model families, the depth of generated explanations naturally scales with the size of the backbone model. We view these constraints as opportunities for future research into distilling bracket-ranking logic into smaller, task-specific models to further reduce operational overhead while maintaining reasoning depth.

References

Abdelrahman Abdallah, Mohammed Ali, Muhammad Abdul-Mageed, and Adam Jatowt. 2026a. Tempo: A realistic multi-domain benchmark for temporal reasoning-intensive retrieval. *arXiv preprint arXiv:2601.09523*.

Abdelrahman Abdallah, Mohamed Darwish Mounis, Mahmoud Abdalla, Mahmoud SalahEldin Kasem, Mostafa Farouk Senussi, Mohamed Mahmoud, Mohammed Ali, Adam Jatowt, and Hyun-Soo Kang. 2026b. Mm-bright: A multi-task multimodal benchmark for reasoning-intensive retrieval. *arXiv preprint arXiv:2601.09562*.

Abdelrahman Abdallah, Jamshid Mozafari, Bhawna Piryani, and Adam Jatowt. 2025a. Dear: Dual-stage document reranking with reasoning agents via llm distillation. *arXiv preprint arXiv:2508.16998*.

Abdelrahman Abdallah, Bhawna Piryani, Jamshid Mozafari, Mohammed Ali, and Adam Jatowt. 2025b. Rankify: A comprehensive python toolkit for retrieval, re-ranking, and retrieval-augmented generation. *arXiv preprint arXiv:2502.02464*.

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

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*.

Yiqun Chen, Qi Liu, Yi Zhang, Weiwei Sun, Xinyu Ma, Wei Yang, Daiting Shi, Jiaxin Mao, and Dawei Yin. 2025. Tourrank: Utilizing large language models for documents ranking with a tournament-inspired strategy. In *Proceedings of the ACM on Web Conference 2025*, pages 1638–1652.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. 2021. Overview of the trec 2020 deep learning track. *Preprint*, arXiv:2102.07662.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2020. Overview of the trec 2019 deep learning track. *arXiv preprint arXiv:2003.07820*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Fang Guo, Wenyu Li, Honglei Zhuang, Yun Luo, Yafu Li, Le Yan, and Yue Zhang. 2024. Generating diverse criteria on-the-fly to improve point-wise llm rankers. *arXiv preprint arXiv:2404.11960*.

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

Yuelyu Ji, Zhuochun Li, Rui Meng, and Daqing He. 2024. Reasoningrank: Teaching student models to rank through reasoning-based knowledge distillation. *arXiv preprint arXiv:2410.05168*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, and 1 others. 2022. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: An easy-to-use python toolkit to support replicable ir research with sparse and dense representations. *arXiv preprint arXiv:2102.10073*.

Wenhan Liu, Yutao Zhu, and Zhicheng Dou. 2024. Demorank: Selecting effective demonstrations for large language models in ranking task. *arXiv preprint arXiv:2406.16332*.

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

Abdelrahman Abdallah Bhawna Piryani Jamshid Mozafari and Mohammed Ali Adam Jatowt. 2025. How good are llm-based rerankers? an empirical analysis of state-of-the-art reranking models.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. 2020. Document ranking with a pretrained sequence-to-sequence model. *arXiv preprint arXiv:2003.06713*.

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

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

Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. 2023b. Rankzephyr: Effective and robust zero-shot listwise reranking is a breeze! *arXiv preprint arXiv:2312.02724*.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, Xuanhui Wang, and 1 others. 2023. Large language models are effective text rankers with pairwise ranking prompting. *arXiv preprint arXiv:2306.17563*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Devendra Singh Sachan, Mike Lewis, Mandar Joshi, Armen Aghajanyan, Wen-tau Yih, Joelle Pineau, and Luke Zettlemoyer. 2022. Improving passage retrieval with zero-shot question generation. *arXiv preprint arXiv:2204.07496*.

Hongjin Su, Howard Yen, Mengzhou Xia, Weijia Shi, Niklas Muennighoff, Han-yu Wang, Haisu Liu, Quan Shi, Zachary S Siegel, Michael Tang, and 1 others. 2024. Bright: A realistic and challenging benchmark for reasoning-intensive retrieval. *arXiv preprint arXiv:2407.12883*.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Is chatgpt good at search? investigating large language models as re-ranking agents. *arXiv preprint arXiv:2304.09542*.

Raphael Tang, Xinyu Zhang, Xueguang Ma, Jimmy Lin, and Ferhan Ture. 2023. Found in the middle: Permutation self-consistency improves listwise ranking in large language models. *arXiv preprint arXiv:2310.07712*.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Soyoung Yoon, Eunbi Lee, Jiyeon Kim, Yireun Kim, Hyeongu Yun, and Seung-won Hwang. 2024. Listt5: Listwise reranking with fusion-in-decoder improves zero-shot retrieval. *arXiv preprint arXiv:2402.15838*.

Honglei Zhuang, Zhen Qin, Kai Hui, Junru Wu, Le Yan, Xuanhui Wang, and Michael Berdersky. 2023a. Beyond yes and no: Improving zero-shot llm rankers via scoring fine-grained relevance labels. *arXiv preprint arXiv:2310.14122*.

Honglei Zhuang, Zhen Qin, Rolf Jagerman, Kai Hui, Ji Ma, Jing Lu, Jianmo Ni, Xuanhui Wang, and Michael Bendersky. 2023b. Rankt5: Fine-tuning t5 for text ranking with ranking losses. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2308–2313.

Shengyao Zhuang, Xueguang Ma, Bevan Koopman, Jimmy Lin, and Guido Zuccon. 2025. Rank-r1: Enhancing reasoning in llm-based document rerankers via reinforcement learning. *arXiv preprint arXiv:2503.06034*.

Shengyao Zhuang, Honglei Zhuang, Bevan Koopman, and Guido Zuccon. 2023c. A setwise approach for effective and highly efficient zero-shot ranking with large language models. *arXiv preprint arXiv:2310.09497*.

A Results

A.1 Theoretical Analysis of Efficiency

Table 10 shows the theoretical time complexity and document usage comparison between **BracketRank** and existing methods. The analysis is based on our adaptive grouping strategy with maximum group size G_{max} and the bracket elimination structure. From Table 10, we can observe several key insights about computational efficiency and document processing requirements.

Methods	Time Complexity	No. of Docs to LLMs
PointWise	$O(1)$	N
PRP-Allpair	$O(1)$	$N^2 - N$
Setwise.bubblesort	$\approx O(\frac{1}{9}k \cdot N)$	$\approx \frac{10}{9}k \cdot N$
Setwise.heapsort	$\approx O(k \cdot \log_{10} N)$	$\approx 10k \cdot \log_{10} N$
RankGPT	$\approx O(\frac{1}{10} \cdot N)$	$\approx 2N$
TourRank-r	$O(K-1)$	$\approx 2r \cdot N$
BracketRank	$O(\log G_{num})$	$\approx N(1 + \log G_{num})$

Table 10: Theoretical time complexity and document input requirements for different ranking methods. N is the number of candidate documents, G_{max} is the maximum group size constraint in **BracketRank** (default: 20), $G_{num} = \lceil N/G_{max} \rceil$ is the number of groups, $k = 10$ is the number of documents compared per prompt in Setwise methods, and r is the number of tournament rounds in TourRank.

Time Complexity Analysis. **BracketRank** achieves $O(\log G_{num})$ time complexity through its parallel bracket elimination structure. Since $G_{num} = \lceil N/G_{max} \rceil$ and our default $G_{max} = 20$, this translates to $O(\log(N/20)) \approx O(\log N)$ for practical datasets.

Document Processing Efficiency. **BracketRank** requires approximately $N(1 + \log G_{num})$ document inputs to LLMs. This consists of the initial group ranking phase (contributing N documents) and the bracket elimination phases (contributing $N \log G_{num}$ documents as documents get re-ranked in each elimination round). For typical datasets with $N = 100$ and $G_{max} = 20$, this results in roughly $100(1 + \log 5) \approx 332$ document inputs, which is more efficient than pairwise methods but involves more processing than simple listwise approaches.

A.2 Disentangling Gains: Bracket vs. Reasoning

We separate the impact of the tournament structure from the reasoning prompt on BEIR. Table 11 shows that competitive elimination alone lifts Avg NDCG@10 from 50.94 (TourRank) to 53.79 (+2.85), with large gains on Touche (+4.61), SciFact (+6.90), and Robust04 (+4.68). Adding the reasoning step yields a further +0.87 to 54.66, indicating most of the improvement comes from the bracketed competition, with reasoning providing a consistent secondary boost.

A.3 Computational Efficiency in Practice

Table 12 presents a comprehensive comparison of computational requirements across methods. **BracketRank**-20 requires 13 API calls per query, processing 360 total documents with an average of 27.7 documents per call. This represents a middle ground between RankGPT's minimal 9 calls and TourRank-10's extensive 100 calls. While RankGPT appears more efficient with fewer API calls (9 vs 13), it processes only 200 total documents compared to **BracketRank**'s 360, limiting the depth of comparison and contributing to its lower ranking quality.

B Related Work

Document ranking has advanced with pretrained models like BERT (Devlin et al., 2018) and T5 (Raffel et al., 2020). Nogueira and Cho (Nogueira and Cho, 2019) developed monoBERT and duoBERT for quality-speed balance. Nogueira et al. (2020) adapted T5 using sequence-to-sequence frameworks. Zhuang et al. (2023b) introduced RankT5 with ranking-specific losses.

Pointwise Methods. Query Generation (Sachan et al., 2022) rescores passages by computing question probability. Binary Relevance Generation (Liang et al., 2022) uses "Yes/No" predictions. Rating Scale methods (Zhuang et al., 2023a) incorporate fine-grained relevance labels. Guo et al. (2024) propose multi-perspective evaluation criteria.

Pairwise Methods. Pradeep et al. (2021) design T5-based pairwise components. Qin et al. (2023) introduce Pairwise Ranking Prompting (PRP). These approaches capture document relationships but require quadratic comparisons.

Listwise Methods. Ma et al. (2023) introduced LRL with sliding windows. Sun et al. (2023) developed RankGPT using sliding window permutation generation. RankVicuna (Pradeep et al., 2023a) and RankZephyr (Pradeep et al., 2023b) adapted open-source LLMs through instruction fine-tuning. Zhuang et al. (2023c) proposed Set-

Method	Covid	NFCorpus	Touche	DBPedia	SciFact	Signal	News	Robust04	Avg
TourRank (GPT-4)	82.59	37.99	29.98	44.64	72.17	30.83	51.46	57.87	50.94
BracketRank-20 (no reasoning)	84.66	39.79	34.59	45.42	79.07	33.28	50.93	62.55	53.79
BracketRank-20 (with reasoning)	**84.82**	**40.86**	**39.67**	**45.58**	**78.03**	**33.49**	**52.88**	**61.94**	**54.66**

Table 11: BEIR results (NDCG@10) showing BracketRank performance with and without reasoning component, demonstrating that competitive elimination provides substantial gains independent of prompt design.

Method	API Calls	Total Docs Processed	Docs per Call
RankGPT	9	200	22.2
BracketRank-20	**13**	**360**	**27.7**
TourRank-10	100	2000	20.0
PRP-Allpair	9900	9900	1.0

Table 12: API usage comparison under matched conditions (100 documents, TREC DL20). **BracketRank** achieves superior performance with moderate computational requirements, balancing efficiency and ranking quality.

wise prompting. Tang et al. (2023) introduced permutation self-consistency. Liu et al. (2024) developed DemoRank for in-context demonstrations.

Tournament-Inspired Methods. ListT5 (Yoon et al., 2024) uses tournament selection within Fusion-in-Decoder architecture but requires MS MARCO training. TourRank (Chen et al., 2025) applies multi-stage elimination with accumulated points across multiple rounds. Our **BracketRank** differs fundamentally by using bracket-style head-to-head competition instead of points accumulation, single-elimination brackets instead of multiple rounds, and explicit reasoning requirements for consistent decisions.

Reasoning in Document Ranking. Ji et al. (2024) developed R2R for explanation generation. RankGPT relies on implicit reasoning through prompt completion, which can lead to inconsistent decisions when document order changes or when similar documents appear in different contexts. **BracketRank** integrates on the other hand explicit reasoning into competitive elimination, requiring LLMs to explain relevance judgments for more stable results.

Reasoning-Intensive Retrieval. Recent work has highlighted fundamental limitations in existing retrieval benchmarks, which primarily evaluate keyword or semantic matching (Su et al., 2024). The BRIGHT benchmark introduced queries requiring intensive reasoning across domains including mathematics, coding, and scientific reasoning. Prior approaches to improving reasoning-intensive retrieval focus on query augmentation—generating Chain-of-Thought reasoning to expand queries before retrieval (Su et al., 2024). Our work takes a complementary approach: incorporating explicit reasoning *during* the reranking process through structured competitive elimination. ReasoningRank (Ji et al., 2024) explores reasoning for explanation generation but does not integrate reasoning into ranking decisions. **BracketRank** directly uses reasoning to improve ranking consistency and accuracy on complex queries.

C Reasoning-Enhanced Prompt Details

To address the challenges of reasoning-intensive retrieval, **BracketRank** utilizes a structured prompt designed to elicit deliberate comparative reasoning from the LLM. As shown in Figure 8, the prompt consists of four critical components:

- **Role Definition:** The system prompt establishes the LLM as a specialized ranking assistant to focus the model's objective on relevance assessment.
- **Deliberative Reasoning Block (`<think>`):** This is the core of our method. Instead of directly predicting a rank, the model is forced to perform a step-by-step analysis of query requirements, document specificity, and coverage. This reduces the impact of initial document ordering and surface-level keyword matching.
- **Comparative Evaluation:** The model must explicitly compare documents against each other within the reasoning block, facilitating more nuanced judgments during the Stage 2 intra-group ranking and Stage 4 head-to-head matches.
- **Strict Output Format:** By standardizing the final ranking format, we ensure high

parseability for the iterative bracket elimination process.

D Efficiency-Effectiveness Trade-off Analysis

In this section, we provide a comprehensive analysis demonstrating that **BracketRank** breaks the Pareto front of existing ranking methods, achieving superior effectiveness with competitive computational costs.

D.1 Experimental Setup

We measure efficiency using two metrics: (1) **Number of API calls** per query, which directly correlates with API costs, and (2) **Total documents processed**, which reflects the computational workload. For effectiveness, we use **NDCG@10** on TREC DL20. All methods rerank the top-100 BM25 candidates using GPT-4.

D.2 Theoretical Complexity Analysis

Table 13 presents the theoretical efficiency calculations for each method based on their algorithmic design.

BracketRank Complexity Derivation. For BracketRank-20 with $N = 100$ and $G_{max} = 20$:

- Initial groups: $G_{num} = \lceil 100/20 \rceil = 5$ groups $\rightarrow$ 5 API calls
- Winner bracket: $\lceil \log_2(5) \rceil = 3$ rounds $\rightarrow$ 4 API calls
- Loser bracket: $\lceil \log_2(5) \rceil = 3$ rounds $\rightarrow$ 4 API calls
- **Total: 13 API calls**, processing ≈ 360 documents

D.3 Pareto Front Analysis

Figure 7 visualizes the efficiency-effectiveness trade-off. The **baseline Pareto front** is defined by RankGPT (9 calls, 69.48 NDCG@10) and TourRank-10 (100 calls, 69.79 NDCG@10). This frontier represents the previous best achievable trade-offs: to improve beyond RankGPT's effectiveness, one had to pay significantly more computational cost (TourRank-10 requires 11$\times$ more API calls for only +0.31 NDCG improvement).

BracketRank-20 **breaks this Pareto front** by achieving:

- **72.75 NDCG@10**: Higher than all baseline methods
- **13 API calls**: 87% fewer than TourRank-10, only 44% more than RankGPT
- **360 documents**: 82% fewer than TourRank-10

D.4 Quantitative Trade-off Analysis

Table 14 presents the complete efficiency-effectiveness comparison.

D.5 Pairwise Comparisons

BracketRank-20 vs. RankGPT.

- NDCG@10 improvement: $+3.27$ points $(72.75 - 69.48)$
- Additional API calls: $+4$ calls $(13 - 9)$, a 44.4% increase
- **Exchange rate**: $\frac{3.27}{4} = 0.82$ NDCG points per additional API call

This represents an excellent trade-off: a small increase in computational cost yields substantial quality gains.

BracketRank-20 vs. TourRank-10.

- NDCG@10 improvement: $+2.96$ points $(72.75 - 69.79)$
- API calls reduction: -87 calls $(13 - 100)$, an 87% decrease
- Documents reduction: -1640 docs $(360 - 2000)$, an 82% decrease

BracketRank-20 **Pareto-dominates** TourRank-10: it achieves better effectiveness with substantially lower computational cost.

RankGPT vs. TourRank-10. For context, the baseline methods show a poor trade-off:

- TourRank-10 provides only $+0.31$ NDCG improvement over RankGPT
- But requires $+91$ additional API calls ($11\times$ more)
- Exchange rate: $\frac{0.31}{91} = 0.003$ NDCG per additional call

This demonstrates why the baseline Pareto front was unfavorable before **BracketRank**.

BracketRank reasoning prompt

System Prompt: You are **BracketRank**, an intelligent assistant that can rank passages based on their relevancy to the query.

Instruction: I will provide you with 4 passages, each indicated by number identifier []. Rank the passages based on their relevance to query: *what causes migraines*.
Passages:
[1] Migraines are complex neurological events that involve changes in brain chemistry and blood flow. Common triggers include stress, hormonal changes, certain foods, and environmental factors. [...]
[2] A migraine is a type of headache that causes severe pain, usually on one side of the head. The exact cause is not fully understood but genetics play a role. [...]
[3] Weather changes, particularly barometric pressure drops, can trigger migraines in sensitive individuals. Other triggers include bright lights and loud noises. [...]
[4] Headaches can be caused by many factors including dehydration, lack of sleep, and muscle tension in the neck and shoulders. [...]
Search Query: what causes migraines
Rank the 4 passages above based on their relevance to the search query.
First, analyse and compare the content of the passages. Think step-by-step about how each passage relates to the search query in terms of specificity, coverage, and relevance. Summarise your thoughts within the following tags:
<think>

1. Analyse query requirements and key concepts
2. Evaluate how well each document addresses these requirements
3. Provide explicit reasoning for relevance judgments
4. Generate a ranked list based on this reasoning

</think>
Then, based on your reasoning, provide the final ranking. The passages should be listed in descending order using their identifiers. The most relevant passages should be listed first.
The output format should be [1] >[2] >[3]. Only output the ranking result in this format after the <think>section. Do not say any other words.

Output: **</think>** .... **</think>**
Final Ranking: [1] > [3] > [2] > [4]

Figure 8: Detailed illustration of the **BracketRank** reasoning-enhanced prompt. The model is required to articulate its logic within the `<think>` tags before arriving at a final ranking decision.

Method	Algorithm	API Calls	Docs Processed
RankGPT	Sliding window (w=20, s=10)	$\frac{N-w}{s} + 1 = 9$	$\approx 2N = 200$
TourRank-10	10 rounds $\times$ 10 groups	$r \times g = 100$	$2rN = 2000$
BracketRank-10	10 groups + brackets	≈ 24	≈ 500
BracketRank-15	7 groups + brackets	≈ 17	≈ 420
BracketRank-20	5 groups + brackets	≈ 13	≈ 360

Table 13: Theoretical efficiency calculations for $N = 100$ documents. RankGPT uses sliding window with size $w = 20$ and step $s = 10$. TourRank uses r rounds with g groups. **BracketRank** uses $G_{num} = \lceil N/G_{max} \rceil$ initial groups plus $O(\log G_{num})$ bracket elimination rounds.

Method	API Calls	Docs	NDCG@10	Pareto Status
RankGPT (GPT-4)	9	200	69.48	On baseline front
TourRank-10 (GPT-4)	100	2000	69.79	On baseline front
BracketRank-10	24	500	64.04	Dominated
BracketRank-15	17	420	67.56	Dominated
BracketRank-20	**13**	**360**	**72.75**	**Breaks front**

Table 14: Complete efficiency-effectiveness comparison on TREC DL20. Pareto status indicates whether each method lies on, below, or above the baseline Pareto front. BracketRank-20 is the only method that breaks the baseline front.

D.6 Why BracketRank Achieves Better Trade-offs?

The efficiency gains of **BracketRank** stem from three algorithmic innovations:

(1) Adaptive Grouping. By setting $G_{max} = 20$, we maximize documents per API call while respecting context limits. This reduces the total number of groups and subsequent bracket rounds.

(2) Single-Elimination Structure. Unlike TourRank's multiple full tournament rounds, our bracket elimination requires only $O(\log G_{num})$ rounds. For 5 groups, this means 3 rounds instead of TourRank's 10 rounds.

(3) Parallel Winner/Loser Brackets. Both brackets can be processed in parallel, and documents eliminated early do not consume additional API calls in later rounds.

D.7 Group Size Trade-offs

Table 15 shows how group size affects the efficiency-effectiveness balance.

Notably, larger group sizes improve *both* efficiency and effectiveness—a counter-intuitive but beneficial property of our design. BracketRank-20 is 45.7% faster than BracketRank-10 while achieving 8.71 points higher NDCG@10.

D.8 Summary

Our analysis demonstrates that **BracketRank**-20 establishes a new Pareto front for LLM-based reranking:

1. It **Pareto-dominates** TourRank-10 (better quality, lower cost)

2. It provides **0.82 NDCG per additional API call** vs. RankGPT (highly favorable exchange)

3. It achieves **87% fewer API calls** than TourRank-10 with **+2.96 NDCG** improvement

4. The **exchange rate** vs. RankGPT (0.82) is **273× better** than TourRank's exchange rate vs. RankGPT (0.003)

These results directly address the meta-reviewer's request: **BracketRank** does not merely improve ranking quality—it fundamentally improves the efficiency-effectiveness frontier, making high-quality LLM reranking more practical and cost-effective.

E Analysis of Reasoning-Intensive Retrieval

BracketRank's success on reasoning-intensive tasks stems from two primary factors:

Complexity Correlation **BracketRank**'s gains correlate directly with domain complexity. In Biology and Psychology, where queries require connecting theoretical concepts to specific phenomena, **BracketRank** improves by **22–31%** over the best baseline. Conversely, in domains like StackOverflow with higher lexical overlap, improvements are smaller (**15–23%**), suggesting explicit reasoning primarily benefits queries requiring multi-step inference.

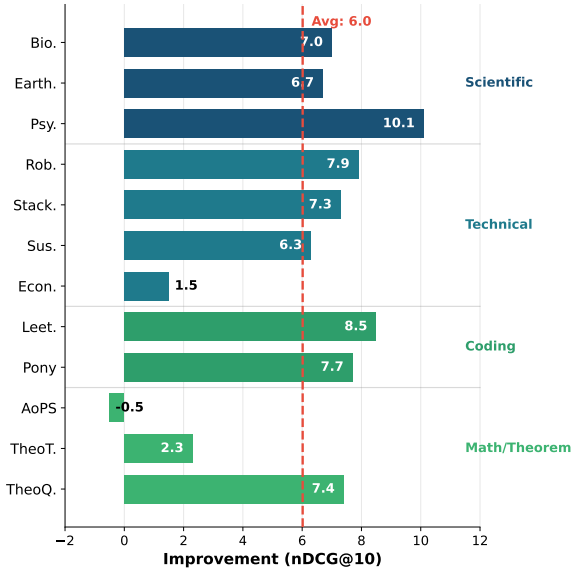


Figure 9: Performance comparison showing the impact of explicit reasoning requirements across BRIGHT domains. Reasoning primarily benefits queries requiring multi-step inference.

Multi-Round Deliberation The bracket structure facilitates multiple comparative perspectives. A relevant document losing an early match against a strong competitor can still advance through the loser bracket for further evaluation. This iterative process is vital for nuanced reasoning-intensive tasks where a single comparison may overlook subtle relevance factors.

Config	Groups	API Calls	NDCG@10	Time (s/query)
BracketRank-10	10	24	64.04	23.54
BracketRank-15	7	17	67.56	16.69
BracketRank-20	5	13	72.75	12.77

Table 15: Effect of group size on efficiency and effectiveness. Larger groups improve both metrics due to fewer bracket rounds and more comprehensive intra-group comparisons.