

Tree Reranking from Existing Retrieval Features

nullreference

Abstract

Hybrid retrieval already computes lexical scores, semantic scores, and fusion scores for its candidates. We test whether a tree reranker can use these saved signals to improve their ordering. A LightGBM LambdaMART model reranks the same 100 candidates per query as BM25, reciprocal rank fusion (RRF), and Distribution-Based Score Fusion (DBSF). On held-out SciFact and FiQA queries, the full model scores higher than DBSF on both datasets. Its differences from BM25 are small and uncertain. Most of the model’s split gain comes from DBSF and BM25, while embedding features contribute a smaller share. This supports a narrow claim: trees can improve one hybrid ranking from features already in its retrieval output, but these results do not show an advantage over BM25 or establish gains for other retrieval systems.

1 Question

A retrieval system often computes more than its final ranking uses. A hybrid pass can expose a BM25 score, a dense similarity, a fusion score, embeddings, and other candidate-level signals. A tree can combine these values without recomputing them. The question is whether that extra ranking step improves the ordering enough to justify using it.

This study tests one specific case. We compare BM25, dense similarity, RRF, DBSF, and LightGBM LambdaMART on identical exported hybrid candidate sets. This isolates reranking quality from candidate generation. It does not compare retrieval systems or test a BM25-only candidate pool.

2 Method

The data contain 100 candidates per query from the SciFact and FiQA hybrid retrieval exports. We split query IDs within each dataset into 70% training, 15% validation, and 15% test sets with seed 0. The model trains jointly on the two training sets. Validation nDCG@10 controls early stopping. The test sets remain separate by dataset.

For each candidate, the full model receives 1,031 values: the 256-dimensional query and document embeddings, their elementwise product and absolute difference, cosine similarity, BM25 score, dense score, RRF score, DBSF score, query-term coverage, and a retrieval-source indicator. A scalar ablation uses only the seven scalar signals. Both models use LightGBM 4.6.0 LambdaMART with 31 leaves, a 0.05 learning rate, feature and row fractions of 0.9, a 300-round maximum, and early stopping after 20 rounds without validation improvement [3, 4].

All methods reorder the same candidates. BM25 and dense baselines sort by their exported scores. RRF uses the ranks within each returned retrieval list, with $k = 60$, and gives no contribution when a list did not return a candidate. DBSF normalizes each returned list by its mean and sample standard deviation using the three-standard-deviation interval, then adds the normalized scores. A missing list contributes zero [2, 1].

We report candidate-conditioned nDCG@10 on every held-out query. Queries with no relevant candidate receive zero. Paired 95% confidence intervals use 10,000 query bootstrap resamples. This metric evaluates ordering within the exported candidate pool. It does not measure recall outside that pool.

3 Results

The full tree scores 0.736 on SciFact and 0.285 on FiQA. It exceeds DBSF by 0.043 on each dataset. The paired 95% intervals for those differences are [0.015, 0.075] on SciFact and [0.032, 0.055] on FiQA. The differences from BM25 are smaller: 0.018 on SciFact and 0.003 on FiQA. Both intervals include zero, so this split does not establish that the full tree improves on BM25. The scalar tree also does not clearly improve on BM25.

Table 1: Candidate-conditioned test nDCG@10. Each dataset has a separate query-level test set.

Dataset (n)	BM25	Dense	RRF	DBSF	Scalar tree	Full tree
SciFact (122)	0.719	0.478	0.629	0.693	0.735	0.736
FiQA (826)	0.282	0.127	0.235	0.242	0.276	0.285

The full model’s split gain is concentrated in DBSF (49%), BM25 (13%), and RRF (11%). Embedding features account for 20% in total. These values are descriptive feature importance, not causal contributions. Correlated features can share or displace split gain.

4 Limits

This result applies to one exported hybrid candidate pool and one query split for each of two datasets. Candidate-conditioned nDCG hides relevant documents that retrieval did not return. We did not test a BM25-only retrieval pool, additional retrievers, repeated splits, or serving latency. The main result is therefore limited: this tree improved on DBSF within these candidate pools, but we found no reliable improvement over BM25.

References

- [1] Qdrant. Hybrid Queries: Distribution-Based Score Fusion. <https://qdrant.tech/documentation/search/hybrid-queries/>.
- [2] G. V. Cormack, C. L. A. Clarke, and S. Buettcher. Reciprocal Rank Fusion Outperforms Condorcet and Individual Rank Learning Methods. In *Proceedings of SIGIR*, 2009. <https://doi.org/10.1145/1571941.1572114>.
- [3] G. Ke et al. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Advances in Neural Information Processing Systems 30*, 2017. <https://proceedings.neurips.cc/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html>.
- [4] C. J. C. Burges. From RankNet to LambdaRank to LambdaMART: An Overview. Microsoft Research Technical Report MSR-TR-2010-82, 2010. <https://www.microsoft.com/en-us/research/publication/from-ranknet-to-lambdarank-to-lambdamart-an-overview/>.