ReBOL: Retrieval via Bayesian Optimization with Batched LLM Relevance Observations and Query Reformulation

Anton Korikov
University of Toronto
korikov@mie.utoronto.ca

Scott Sanner
University of Toronto
ssanner@mie.utoronto.ca

Abstract

LLM-reranking is limited by the top-k documents retrieved by vector similarity, which neither enables contextual query-document token interactions nor captures multimodal relevance distributions. While LLM query reformulation attempts to improve recall by generating improved or additional queries, it is still followed by vector similarity retrieval. We thus propose to address these top-k retrieval stage failures by introducing ReBOL, which 1) uses LLM query reformulations to initialize a multimodal Bayesian Optimization (BO) posterior over document relevance, and 2) iteratively acquires document batches for LLM query-document relevance scoring followed by posterior updates to optimize relevance. After exploring query reformulation and document batch diversification techniques, we evaluate ReBOL against LLM reranker baselines on five BEIR datasets and using two LLMs (Gemini-2.5-Flash-Lite, GPT-5.2). ReBOL consistently achieves higher recall and competitive rankings, for example compared to the best LLM reranker on the Robust04 dataset with 46.5% vs. 35.0% recall@100 and 63.6% vs. 61.2% NDCG@10. We also show that ReBOL can achieve comparable latency to LLM rerankers.

1 Introduction

Given a query q_0, LLM rerankers and relevance classifiers (e.g., Ma et al. (2023); Upadhyay et al. (2024)) are limited to the top-k vector similarity retrieved documents and cannot recover from low-recall retrieval. While a line of LLM query reformulation work (e.g., Wang et al. (2023); Kostric and Balog (2024)) aims to improve recall by generating one or more query reformulations $\{q_1, ..., q_Q\}$, these reformulations are ultimately still processed by vector similarity retrieval. Unfortunately, vector similarity is limited to scoring separately encoded query and document vectors, precluding contextual query-document token interaction. Further, single-vector retrievers use simple scoring functions such as the dot product which induces a unimodal relevance distribution around the query, while relevance is often distributed multi-modally across distinct document clusters (Van Rijsbergen, 2004). Though multi-vector similarity retrievers such as COLBERTv2 (Santhanam et al., 2022) aim to capture multi-modal relevance via late interaction models, they typically rely on simple aggregation functions such as MaxSim over a dot product between separately encoded query and document tokens – limiting their performance against LLMs which enable neural query-document token interactions.

To overcome these failure modes of the top-k retrieval stage of LLM reformulate-retrieve–rerank pipelines, we instead consider interleaving retrieval with LLM relevance observations in a Bayesian optimization (BO) framework, building on recent work (Kim et al., 2026; Liu et al., 2026). First, BO enables us to use the initial query q_0 and its LLM reformulations $\{q_1, ..., q_Q\}$ to initialize multiple relevance peaks in a multimodal posterior over document relevance (Fig. 2b, left). BO then lets us systematically select batches of documents for which to generate high-fidelity LLM relevance labels to update the posterior and optimize relevance (Fig. 2). Two key questions that emerge are (a) how to initialize and seed multimodal BO using LLM query reformulations and (b) how to efficiently acquire informative document batches for LLM labeling and posterior updates. Thus, in this work, we:

- Introduce ReBOL (**Re**trieval via **B**ayesian **O**ptimization with Batched **L**LM Relevance Observations) which initializes a multimodal BO posterior with LLM query reformulations and actively selects document batches for LLM labeling and relevance optimization.

- Explore efficient BO batch acquisition functions, introducing a novel, MMR (Carbonell

Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing, pages 4364–4376
October 24-29, 2026 ©2026 Association for Computational Linguistics

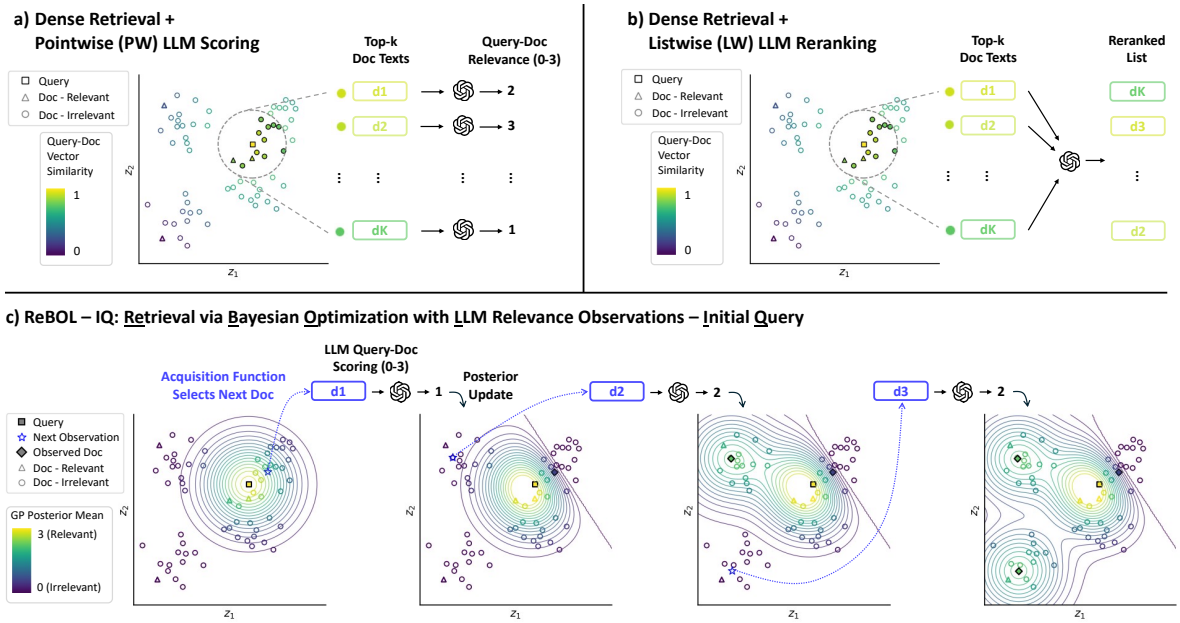

Figure 1: a-b) LLM rankers rely on upstream vector similarity retrieval to reduce a large document collection to a top-k candidate list. c) ReBOL-IQ (**I**nitial **Q**uery) first initializes a GP prior with a high-relevance point at the query embedding (yellow square), then iteratively selects which document to judge next using an acquisition function (blue star), scores its relevance to the query using an LLM, and updates a multimodal GP posterior.

and Goldstein, 1998) inspired batch diversification method

- Evaluate ReBOL on five BEIR (Thakur et al., 2021) datasets against sparse retrieval baselines and LLM rerankers (with/without LLM query reformulation), using two LLMs (Gemini-2.5-Flash-Lite, GPT-5.2) as well as a cross-encoder.

- Demonstrate that ReBOL consistently attains much higher recall and competitive rankings, for instance compared to the best LLM reranker (with LLM query reformulation) on the Robust04 dataset where it achieves 46.5% recall@100 vs. 35.0% and an NDCG@10 of 63.6% vs. 61.2%, respectively.

- Show that ReBOL has comparable latency to LLM rerankers, since LLM latency generally dominates the extra BO overhead.

2 Background and Related Work

2.1 Vector Similarity Retrieval

Vector similarity retrieval relies on an encoder $g(x) = z^x$ which maps some text x to a vector $z^x \in \mathbb{R}^m$. Documents are encoded offline, and given query q, the retriever computes $g(q) = z^q$ and scores it against each z^d using a similarity function $S(\cdot, \cdot) : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ such as the dot product. Sparse retrievers such as BM25 (Robertson and Zaragoza, 2009) use token frequency vectors for lexical matching (Salton et al., 1975), while dense retrievers (e.g., Izacard et al. (2021); Gao and Callan (2021)) use neural encoders to produce dense embeddings that aim to capture semantic similarity. Though multi-vector similarity retrievers such as COLBERTv2 (Khattab and Zaharia, 2020; Santhanam et al., 2022) aim to capture multi-modal relevance via late interaction models, they typically rely on simple aggregation functions such as MaxSim over a dot product between separately encoded query and document tokens, rather than enabling neural interactions between query and document tokens.

2.2 LLM-augmented Retrieval

2.2.1 LLM Query Reformulation

Given an initial query q_0, recent work shows that retrieval can be improved by using an LLM to generate a query reformulation (e.g., Wang et al. (2023); Jagerman et al. (2023); Dhole and Agichtein (2024); Wen et al. (2025)) or multiple reformulations $\{q_1, ..., q_Q\}$ (e.g., (Lin et al., 2021; Kostric and Balog, 2024; Dhole et al., 2024; Korikov et al., 2024)) to be then used for vector sim-

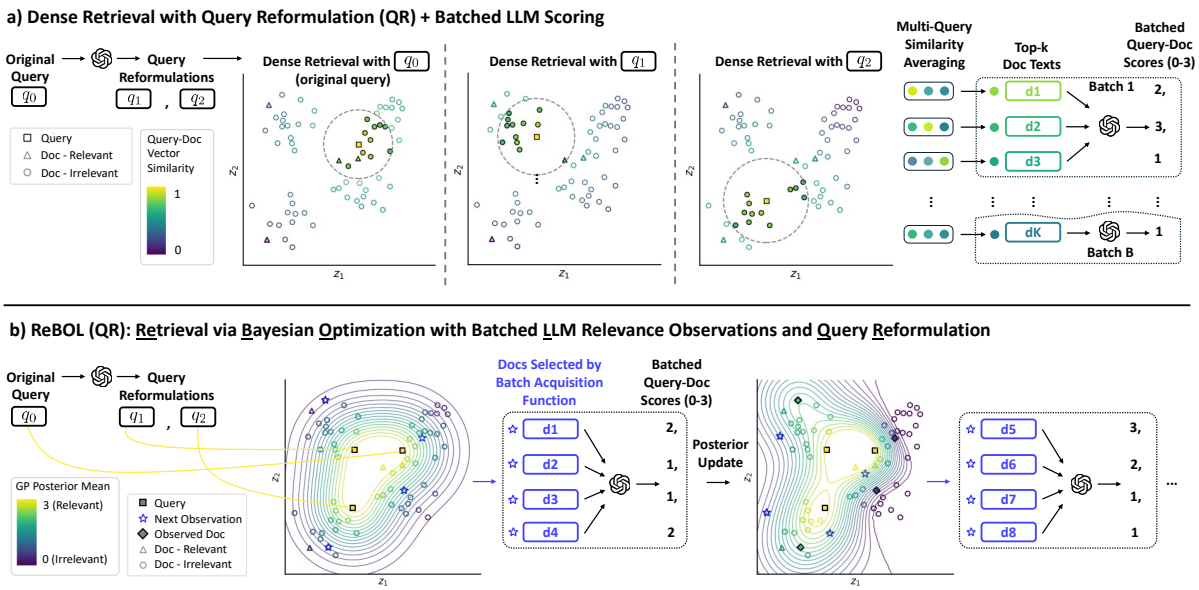

Figure 2: Given q_0, an LLM generates query reformulations q_1 and q_2. a) An example reformulate-retrieve-rerank pipeline, using each $q \in \{q_0, q_1, q_2\}$ for vector similarity retrieval before score averaging and batched top-k LLM scoring b) ReBOL-QR (**Q**uery **R**eformulation) initializes a multimodal GP posterior with peaks at query embeddings $\{z^{q_0}, z^{q_1}, z^{q_2}\}$ followed by active document batch acquisition, batched LLM scoring, and posterior updating.

ilarity retrieval. Multi-query retrieval results are typically aggregated via similarity averaging (e.g., Fig 2a) or ranked list fusion.

2.2.2 LLM Relative Relevance Judgment

Given q_0 and a top-k document list, listwise (LW) and pairwise rerankers judge relative relevance between documents. Specifically, LW methods prompt an LLM to reorder a document list by descending relevance to q_0 (e.g., Fig 1b). The most basic LW variation (Ma et al., 2023) processes all k documents in a single LLM call, but including too many candidates can degrade performance, so stronger methods use an upwards sliding window strategy (Ma et al., 2023; Sun et al., 2023; Pradeep et al., 2023a,b). Pairwise LLM rerankers (Qin et al., 2024; Liu et al., 2024) limit the documents per LLM relative judgment call even further to only two, often performing strongly — but unfortunately require a quadratic number of LLM calls so are not included in our experiments.

2.2.3 LLM Absolute Relevance Scoring

Generating real-valued LLM query-document relevance scores $s_{q,d} \in \mathbb{R}$ is often called pointwise (PW) reranking, and is typically done using some rubric such as the 0-3 UMBRELA labels below (Upadhyay et al., 2024):

- **3:** The passage is dedicated to the query and contains the exact answer.
- **2:** The passage has some answer for the query, but the answer may be a bit unclear, or hidden amongst extraneous information.
- **1:** The passage seems related to the query but does not answer it.
- **0:** The passage has nothing to do with the query.

Documents can be LLM-scored one-by-one (Sachan et al., 2022; Upadhyay et al., 2024; Törnberg, 2024) or as a batch (Korikov et al., 2025), with the latter often increasing efficiency and sometimes performance as it enables interactions between documents.

2.3 Bayesian Optimization

In BO (Garnett, 2023), we start with a real-valued function over some domain $\mathcal{X}$; $f : \mathcal{X} \to \mathbb{R}$ which we assume is distributed according to some prior $p(f(x)|x)$, and our goal is to systematically search for $x^* \in \arg\max_{x \in \mathcal{X}} f(x)$. To gain information about f, which may be a black-box function, we have some mechanism to observe a $y \in \mathbb{R}$ at an arbitrary point x. We assume that y is distributed according to an observation model $y \sim p(y|x, f(x))$, with a common observation model being Gaussian noise about an $f(x)$ mean:

$$p(y|x, f(x), \sigma_n) = \mathcal{N}(y; f(x), \sigma_n). \quad (1)$$

We also assume conditional independence between observations $\mathbf{y} = [y^1, ..., y^t]$ at $\mathbf{x} = [x^1, ..., x^t]$:

$$p(\mathbf{y}|\mathbf{x}, f(\mathbf{x})) = \prod_{i}^{t} p(y^i|x^i, f(x^i)), \quad (2)$$

and Bayes theorem defines our new posterior as $p(f(\mathbf{x})|\mathcal{D}^t) \propto p(f(\mathbf{x})|\mathbf{x})p(\mathbf{y}|\mathbf{x}, f(\mathbf{x}))$ where $\mathcal{D}^t = (\mathbf{x}, \mathbf{y})$. At any time t, the posterior can be used to predict f (e.g., Sec. 2.3.1, Eq (4),(5)) and inform an acquisition function $\alpha : \mathcal{X} \to \mathbb{R}$ to select the next observation location x^{t+1} as $\arg\max_x \alpha(x|\mathcal{D}^t)$ (Sec. 3.5).

2.3.1 Gaussian Processes

Since it enables efficient closed-form inference, a very common prior is the Gaussian process (GP), which for a zero mean is defined as $p(f) = GP(0, k)$, where $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_0^+$ is called the kernel. For any two points, $k(x, x')$ represents the covariance between $f(x)$ and $f(x')$, and a common choice is the RBF kernel

$$k(x, x') = \sigma_s^2 \exp\left(-\frac{\|x - x'\|^2}{2\ell^2}\right), \quad (3)$$

where ℓ is the length-scale controlling how quickly correlations decay, and σ_s^2 is the signal variance.

Given t observations $\mathbf{x}, \mathbf{y}$ following the Gaussian noise model in Eq. (1), the GP predictive posterior used to predict $f(x')$ at any $x' \in \mathcal{X}$ is $p(f(x')|\mathbf{x}, \mathbf{y}, x') = \mathcal{N}(\mu[f(x')], \sigma^2[f(x')])$ with:

$$\mu[f(x')] = \mathbf{k}^T[\mathbf{K} + \sigma_n^2 I]^{-1}\mathbf{y}, \quad (4)$$

$$\sigma^2[f(x')] = k(x', x') - \mathbf{k}^T[\mathbf{K} + \sigma_n^2 I]^{-1}\mathbf{k}, \quad (5)$$

where $\mathbf{K}_{ij} = k(x_i, x_j)$, and $\mathbf{k}_i = k(x', x_i)$ for $i, j \in [1, ..., t]$.

3 Methodology

To address the top-k performance limits imposed by vector similarity retrieval on LLM query reformulation and reranking, we re-frame retrieval as a BO task, introducing ReBOL (Fig. 1c & 2b) — which first uses q_0 and any LLM reformulations $\{q_1, ..., q_Q\}$ as observations to initiate a multimodal GP posterior (Sec. 3.1-3.3), followed by iterative selection of documents with a batch acquisition function (Sec. 3.5), batched LLM query-document relevance scoring (Sec. 3.4), and posterior updating (Sec. 3.6).

3.1 Query-Document Relevance Function

We first define a query-document relevance function $f : \mathbb{R}^m \to \mathbb{R}$ over the embedding space of an m-dimensional dense text encoder $g(x) = z^x$, and let the set of all embedded documents $d \in \mathcal{X}$ be represented by $\mathcal{Z}^{\mathcal{X}} \subset \mathbb{R}^m$. While $f(z)$ could have any form, in this work we find it convenient to bound $f(z) \in [0, s^{\max}]$ where 0 represents irrelevance and $s^{\max} \in \mathbb{R}$ represents maximum relevance (e.g., 3 in the prompt in Sec. 2.2.3). Unlike vector similarity, we assume f can be multimodal, reflecting that relevance is often distributed between multiple clusters (Van Rijsbergen, 2004). While GP mechanics assume f is unbounded, bounded functions are often used in practice (Garnett, 2023).

3.2 Multimodal GP Relevance Prior

To enable efficient inference, we use a GP prior (Sec 2.3.1) over $f(z)$, using a zero mean to model the assumption that most documents are irrelevant for a given q_0. We use a non-linear kernel (e.g. RBF, Eq. (3)) to enable multimodality (e.g., see Figs. 1c & 2b showing distinct relevance peaks). In contrast, vector retrieval with dot product or (normalized) cosine similarity induces unimodal relevance due to its linear scoring function.

3.3 Initial Query and LLM Reformulations

The initial query q_0 provides a natural first observation since it is reasonable to assume maximal relevance at the query embedding, giving the observation pair $(z^1, y^1) = (z^{q_0}, s^{\max})$, as shown on the left of Fig. 1c. Further, LLM-generated reformulations $\{q_1, \ldots, q_Q\}$ can capture different aspects of the query and be incorporated as analogous observations $(z^{i+1}, y^{i+1}) = (z^{q_i}, s^{\max})$ for $i \in [1, \ldots, Q]$, with the goal of inducing multiple GP relevance peaks in the embedding space. Our experiments consider two ReBOL variants: ReBOL-IQ, which uses only the initial query observation, and ReBOL-QR, which incorporates query reformulations.

3.4 LLM Document Relevance Scores

After initialization with query observations, ReBOL iteratively uses an acquisition function (Sec. 3.5) to select a batch of B documents $\mathbf{d}^t = [d^t, ..., d^{t+B-1}]$ at each time step t. An LLM is then used (as per Sec. 2.2.3) to generate relevance scores $\{s_{q_0,d^{t'}}\}$ with $t' \in [t, ..., t + B - 1]$, giving observations at document embeddings $(z^{t'}, y^{t'}) =$

$(z^{d^{t'}}, s_{q_0,d^{t'}})$. Here, batched BO acquisition and LLM scoring can reduce latency and enable the potential benefits of inter-document interactions during LLM scoring (Korikov et al., 2025), as well as provide an additional exploration mechanism via batch diversification, as discussed next.

3.5 Document Acquisition Functions

While there exists a wide range of GP acquisition functions (Garnett, 2023), we explore the following variants, including a novel MMR-style document batch diversification method.

3.5.1 Single Document Acquisition

As per Sec. 2.3, we define a single-document acquisition function $\alpha : \mathcal{Z}^{\mathcal{X}} \to \mathbb{R}$ over document embeddings $\mathcal{Z}^{\mathcal{X}}$, which is used to select the next document d^{t+1} as $\arg\max_{d\in\mathcal{X}} \alpha(z^d|\mathcal{D}^t)$ given observations $\mathcal{D}^t$.

Greedy Our first acquisition function is *greedy*, which simply equals the posterior mean (Eq. (4)), giving $\alpha(z^d|\mathcal{D}^t) = \mu[f(z^d)]$. Greedy is an exploitation-only strategy, meaning that it will prioritize observing documents that are embedded near the query, query reformulations, and any documents with high observed relevance.

UCB The Upper Confidence Bound (UCB) acquisition function balances exploration and exploitation, combining both the posterior mean (Eq. (4)) and uncertainty (Eq. (5)) as

$$\alpha(z^d|\mathcal{D}^t) = \mu[f(z^d)] + \sqrt{\beta}\,\sigma[f(z^d)], \quad (6)$$

encouraging exploration of uncertain regions that may contain highly relevant documents, with β controlling the exploration-exploitation tradeoff.

Random Tested as a maximal exploration baseline, this function selects documents randomly.

3.5.2 Document Batch Acquisition

To select a batch $\mathcal{B}^t$ of documents $\mathbf{d}^t = [d^t, \ldots, d^{t+B-1}]$ at time step t, and given some single-point acquisition function, we explore the following (diversified) batch acquisition styles, considering both BO inspired and Information Retrieval (IR)-inspired variants.

Top-B This no-diversification strategy selects the top-B documents according to the single-point acquisition function, and is the least computationally expensive since it uses only one computation of $\alpha(z^d|\mathcal{D}^t)$ per batch.

Table 1: BEIR (Thakur et al., 2021) dataset statistics for corpus size, # of queries, and mean # of relevant docs.

Dataset	#docs	#qs	#rel d/q
TREC-NEWS	595K	57	19.6
Robust04	528K	249	69.9
TREC-COVID	171K	50	493.5
SciFact	5K	300	1.1
NFCorpus	3.6K	323	38.2

Kriging Believer (KB) Starting from a single-point selection, KB (Ginsbourger et al., 2010) sequentially adds points to $\mathcal{B}^t$ by temporarily assuming the posterior mean (4) as the observation for each selected document, which for uncertainty-aware α such as UCB reduces nearby uncertainty and encourages subsequent selections in other regions. KB requires roughly B times the computation of Top-B since we recompute the posterior B times per batch.

Maximal Marginal Relevance (MMR) Inspired by MMR, a classic information retrieval diversity mechanism (Carbonell and Goldstein, 1998), we introduce a novel MMR-style document batch acquisition function that sequentially builds $\mathcal{B}^t$ by balancing the value of $\alpha(z^d|\mathcal{D}^t)$ for each z^d with its similarity to already selected documents. Starting from $d^t = \arg\max_d \alpha(z^d|\mathcal{D}^t)$, we let

$$d^{t+i} = \arg\max_{d\in\mathcal{X}\setminus\mathcal{B}^t} \big[\, \lambda\, \alpha(z^d|\mathcal{D}^t) - (1-\lambda) \max_{d'\in\mathcal{B}^t} S(z^d, z^{d'}) \big],$$

where $S(\cdot,\cdot)$ is a similarity function (e.g., cosine) and λ controls the diversity level. Unlike KB, the acquisition function is only computed once, followed by $(B-1)|\mathcal{Z}^{\mathcal{X}}|$ similarity computations which are much cheaper than posterior updates.

3.6 ReBOL Relevance Prediction and Summary

At any time step t, the predicted relevance of a document d is given by the GP posterior mean evaluated at the document embedding, $\mu[f(z^d)]$ (Eq. (4)). This allows ReBOL to return ranked documents (with relevance scores) at any stage of the search, making it an anytime retrieval algorithm.

In summary, ReBOL initializes a multimodal relevance posterior using the query and its LLM reformulations, and then iteratively acquires (diverse) document batches for LLM query–document scoring and posterior updates, using a BO framework

that allows LLM relevance signals to guide the search for relevant documents. We next discuss our experiments to investigate whether ReBOL improves on the failures of the top-k retrieval stage of reformulate-retrieve-rerank pipelines.

4 Experimental Setup

We design a set of experiments to compare ReBOL against sparse, dense, and LLM reranker baselines, investigating the effects of LLM query reformulation as well as diversified batched document acquisition and scoring.[1] Specifically, we report recall, NDCG, and latency on the five BEIR (Thakur et al., 2021) datasets in Table 1 — these are the five BEIR benchmarks with the fewest queries per dataset (with the exception of Webis-Touche2020 which was found to be highly biased towards token-matching relevance and BM25 in a reproducibility study by the BEIR authors (Thakur et al., 2024)). LLM temperature is always set to 1.

4.1 ReBOL Implementation Details

Query Reformulation Given initial query q_0, we prompt Gemini-2.5-Flash-Lite to generate four query reformulations to give a total of five queries with q_0, with our exact prompt shown in Fig. 3.

Dense Text Embeddings As our dense encoder $g : \mathcal{X} \to \mathcal{Z}^{\mathcal{X}} \subset \mathbb{R}^m$ we use `all-MiniLM-L6-v2`[2] with $m = 384$ and normalized embeddings.

Relevance Function and LLM Scoring We use the 0-3 UMBRELA (Upadhyay et al., 2024) relevance scale defined in Section 2.2.3 for LLM relevance scoring and to place bounds [0,3] on $f(z)$.

GP Prior For our GP prior, we use a 0 mean and RBF kernel (3) with σ_s, σ_n, and ℓ set to 1.

Acquisition Functions As per Section 3.5, and using a batch size $B = 10$ unless otherwise stated, we test three batch acquisition mechanisms: Top-B, KB, and MMR with $\lambda \in \{0.5, 0.7, 0.9\}$. These batch functions rely on some single-point acquisition function, for which we test greedy, random, and UCB with β set to 1.

Number of Observations In our main experiments, ReBOL observes exactly 100 documents iteratively selected by the acquisition function, while the appendix includes ablation studies with 50 and 200 observations.

4.2 Baseline Implementations

4.2.1 Vector Similarity Retrieval

Dense Retrieval Our dense retrieval baseline uses the same text encoder as in Section 4.1 and dot product similarity. We also test dense retrieval with MMR (Carbonell and Goldstein, 1998) at the same λ values as for ReBOL-MMR in Sec. 4.1.

Dense Retrieval (QR) Our LLM query reformulation dense retrieval baseline uses the same query generation process as for ReBOL in Sec. 4.1. We then follow the retrieval process shown in Fig. 2a, in which $Q + 1$ similarity scores (one for each reformulation plus q_0) are computed using dense retrieval then averaged for the final ranking.

Sparse Retrieval We report several `pyserini`[3] sparse retrieval baselines including BM25-Flat, BM25 Multi-Field (MF), and SPLADE.

4.2.2 LLM Reranking

All LLM rerankers judge the top-k documents from dense retrieval, with k fixed to 100 in our main experiments, matching the number of observations made by ReBOL. The appendix includes ablations with $k = 50$ and $k = 200$.

LW LLM Reranking As per Section 2.2.2, we test two LW variants. The first, *LW-one-call* reranks all k documents in a single LLM call, using the prompt in Fig 4. The second, *LW-window*, uses the same prompt but with an upwards sliding window of W documents — for which we use a step size $\frac{W}{2}$ and $W = 20$ to maintain a comparable number of LLM calls (i.e., 9) for $k = 100$ to ReBOL and PW reranking (10 LLM calls with batch size 10).

(Batched) PW LLM Reranking PW LLM reranking uses the same rubric (Sec. 2.2.3) as ReBOL, and unless otherwise stated, also uses a default batch size of $B = 10$.

4.2.3 ReBOL-Top-k Ablation

We include a version of ReBOL which, after initializing the posterior with query observations, simply makes observations at the dense retriever top-k instead of iteratively using an acquisition function, allowing us to test whether active acquisition is beneficial.

[1] Code is available at: `https://anonymous.4open.science/r/ReBOL-5D56/README.md`

[2] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

[3] https://github.com/castorini/pyserini

Table 2: Recall(R)@100 and NDCG(N)@10 using Gemini-2.5-Flash-Lite for ReBOL with Top-B acquisition ($B = 10$) versus LLM rerankers, including IQ (initial query) and QR (query reformulations added) variants, with * showing 95% significance. The best overall method is **bold** while the best absolute LLM scoring method (PW or ReBOL) is underlined. QR is generally helpful, and ReBOL with greedy or UCB acquisition consistently shows large improvements in recall and competitive rankings.

Method	nfcorpus		scifact		robust		covid		news	
	R@100	N@10	R@100	N@10	R@100	N@10	R@100	N@10	R@100	N@10
BM25 Flat	24.6	32.2	92.5	67.9	37.5	40.7	10.9	59.5	44.7	39.5
BM25 MF	25.0	32.5	90.8	66.5	37.5	40.7	11.4	65.6	42.2	39.8
SPLADE	28.4	34.7	93.5	70.4	38.5	46.8	12.8	72.7	44.1	41.5
COLBERTv2	28.2	34.1	92.5	69.2	37.9	46.4	10.2	74.4	43.4	40.7
Dense (IQ)	30.8	31.6	92.8	64.9	32.7	40.5	9.5	51.8	42.3	38.2
PW (IQ)	30.8	37.7	92.8	70.4	32.7	59.8	9.5	71.3	42.3	45.8
LW-one-call (IQ)	30.8	36.4	92.8	73.0	32.7	51.3	9.5	74.5	42.3	44.1
LW-window (IQ)	30.8	37.6	92.8	74.4	32.7	56.7	9.5	**78.1**	42.3	48.2
Dense (QR)	33.7	33.7	96.6	70.3	35.0	44.2	8.0	45.4	45.7	43.3
PW (QR)	33.7	39.3	96.6	71.0	35.0	61.2	8.0	66.5	45.7	44.0
LW-one-call (QR)	33.7	37.3	96.6	76.1	35.0	52.8	8.0	70.6	45.7	44.0
LW-window (QR)	33.7	39.9	96.6	**76.7**	35.0	58.6	8.0	72.6	45.7	**49.6**
ReBOL-IQ-Rand.	29.9	25.9	91.2	54.5	31.9	38.2	8.0	46.7	40.3	32.8
ReBOL-IQ-Top-k	34.3	38.3	92.8	66.7	38.1	59.5	13.7	75.9	45.6	45.9
ReBOL-IQ-Greedy	36.3	38.8	96.0	69.4	45.1*	61.3	15.3*	74.6	49.8	45.4
ReBOL-IQ-UCB	35.6	38.4	93.7	69.3	44.9*	60.7	15.4*	70.8	52.2	42.8
ReBOL-QR-Rand.	32.8	33.8	96.2	70.0	36.1	47.4	12.7	64.0	47.6	44.6
ReBOL-QR-Top-k	34.3	39.6	93.5	73.3	37.6	59.6	12.9	73.0	42.4	45.8
ReBOL-QR-Greedy	36.1	40.1	**96.7**	73.5	45.6*	**63.6**	**15.6***	76.8	49.9	45.1
ReBOL-QR-UCB	**37.7**	**40.2**	**96.7**	73.5	**46.5***	63.3	**15.6***	75.4	**52.7**	45.7

5 Experimental Results

We first report recall and ranking performance of ReBOL vs. baselines, testing both IQ and QR variants with the simplest top-B batch acquisition method (Tab. 2). We then look at performance and latency results for different batch sizes (1 vs. 10) and various batch acquisition strategies (Table 3-5).

RQ1: How does ReBOL performance compare against LLM reranker baselines? ReBOL with greedy and UCB Top-B acquisition achieves large improvements in recall and competitive NDCG compared to LLM rerankers in Table 2, as well as in ablation studies using different LLM scoring models (GPT-5.2, cross-encoder) and numbers of LLM observations (App. Tables 6-8). Further, comparison to the ReBOL-Top-k ablation (Sec. 4.2.3) shows that active learning improves recall considerably.

RQ2: How helpful is LLM query reformulation for retrieval and ReBOL? As shown in Table 2, QR is helpful for both LLM rerankers and ReBOL, improving recall and NDCG for all datasets except TREC-COVID — perhaps due to its large number of relevant documents per query. Further, QR shows warm start benefits, as indicated by the strong lift of QR on ReBOL-Random, and the ablation with 50 observations (Appendix Table 7) showing greater QR improvements for ReBOL than for the default 100 observations.

RQ3: What are the effects of various batching strategies on performance? The ablation in Table 3 shows that $B = 1$ can do well for ReBOL-QR since it uses the most recent information for each acquisition, but requires a long time per query in LLM calls (e.g., 36s+), while a batch size of $B = 10$ reduces latency significantly (e.g., 5s). Table 4 thus compares different ReBOL batch diversification strategies, showing that MMR typically performs best. MMR also helps dense retrieval, but its benefits are amplified by the BO framework of ReBOL.

RQ4: How does ReBOL compare against LLM rerankers in terms of latency? Table 5 shows that ReBOL with Top-B acquisition does not add much latency compared to LLM rerankers since LLM time dominates. MMR increases latency proportionally to corpus size as it requires roughly $(k - B)|\mathcal{X}|$ similarity evaluations, but is the best performing batching method in Table 4. KB takes

Table 3: Batch size $B = 1$ vs. 10 ablation for ReBOL-QR (Top-B) versus PW Reranking (QR) showing Recall(R)@100 and NDCG(N)@10, as well as mean time per query spent in LLM calls versus in total — both PW and ReBOL see large latency reductions due to batching. ReBOL performs noticeably better with $B = 1$, showing the benefits of more interleaved LLM scoring and BO if time is available.

Method	Batch	covid				news			
		Metric		Time (s)		Metric		Time (s)	
		R@100	N@10	LLM	Total	R@100	N@10	LLM	Total
Dense (QR)	n/a	8.0	45.4	0.8	1.0	45.7	43.3	0.6	1.1
PW (QR)	1	8.0	65.9	38.2	38.4	45.9	44.7	37.3	38.0
ReBOL-QR-Greedy	1	15.6*	79.6	39.4	42.9	51.3	**48.4**	37.6	41.9
ReBOL-QR-UCB	1	**16.4***	**82.4**	39.4	46.9	**52.8**	47.9	39.3	63.7
PW (QR)	10	8.0	66.5	**4.9**	**5.0**	45.9	44.0	**5.5**	**5.9**
ReBOL-QR-Greedy	10	15.6*	76.8	5.4	5.9	49.9	45.1	5.7	6.3
ReBOL-QR-UCB	10	15.6*	75.4	5.4	6.4	52.7	45.7	5.8	8.4

Table 4: Effect of diversification methods, including MMR ($x = \lambda$), Kriging Believer, and Top-B (no diversity).

Method	Diversity	nfcorpus		scifact		robust		covid		news	
		R@100	N@10	R@100	N@10	R@100	N@10	R@100	N@10	R@100	N@10
Dense (QR)	n/a	30.8	31.6	92.8	64.9	32.7	40.5	9.5	51.8	42.3	38.2
	MMR 0.9	33.0	33.6	96.6	70.9	34.4	43.9	8.1	46.2	44.2	41.7
	MMR 0.7	29.2	30.8	94.9	71.1	30.0	40.7	7.8	41.6	36.0	34.8
	MMR 0.5	15.6	21.9	74.5	63.7	9.7	24.7	1.9	26.7	8.2	21.9
ReBOL-QR-Greedy	Top-B	36.1	40.1	**96.7**	73.5	45.6*	63.6	15.6*	76.8	49.9	45.1
	MMR 0.9	**38.6**	41.3	96.3	73.5	46.1*	63.9	15.9*	80.5	50.5	47.4
	MMR 0.7	38.5	**41.6**	96.0	74.2	46.8*	63.8	15.9*	78.7	51.4	45.8
	MMR 0.5	37.4	40.8	96.0	75.0	45.8*	64.2	16.0*	81.7	52.2	47.2
ReBOL-QR-UCB	Top-B	37.7	40.2	**96.7**	73.5	46.5*	63.3	15.6*	75.4	52.7	45.7
	KB	37.6	40.7	96.5	72.2	45.6*	63.7	16.0*	**83.2**	51.7	45.5
	MMR 0.9	37.9	41.1	95.7	73.0	46.7*	64.6	15.9*	77.3	52.4	**49.2**
	MMR 0.7	37.9	41.0	96.2	72.3	**47.1***	**65.2**	**16.1***	78.2	**54.2**	47.5
	MMR 0.5	38.3	41.5	95.7	**75.2**	44.7*	63.8	**16.1***	**83.2**	52.2	48.2

the longest but doesn't generally improve performance.

6 Conclusion

This work addresses the limitations of the top-k retrieval stage of LLM reformulate-retrieve-rerank pipelines by proposing ReBOL. ReBOL reframes retrieval as Bayesian Optimization (BO) and first initializes a multimodal posterior with several LLM query reformulations (QR). It then iteratively uses a (diversified) batch acquisition function to select documents, generate LLM query-document relevance scores, and update the BO posterior to optimize relevance. After introducing new techniques for BO retrieval via LLM-QR, batched LLM relevance judgments, and MMR-style batch diversification, we conduct a range of experiments comparing ReBOL to LLM rerankers and vector retrieval baselines on five datasets. Our experiments indicate that ReBOL is a promising new paradigm for improving retrieval and can effectively leverage query reformulations and diversified batch acquisition.

7 Limitations

Our work includes the following limitations. Firstly, we only perform experiments on five datasets, since LLM experiments are very computationally expensive and our tests require testing a number of configurations, including reranking variants and different acquisition functions for our method. In this regard, another limitation is that we only evaluate two LLMs which are Gemini-2.5-Flash-Lite and GPT-5.2 as well as a cross-encoder (ms-marco-MiniLM-L6-v2) as scoring functions for ReBOL and pointwise reranking. We also only report results for three levels of k (i.e. LLM observation number), specifically 50,100, and 200. Similarly future work should incorporate GP hyperparameter tuning as we only test one set of default hyperparameter values, as well as investigate the

Table 5: Mean time (s) per query spent in LLM calls versus in total (including GP inference), with $B = 10$ for ReBOL and PW. The hardware for all non-LLM steps was a small 4 GB VRAM GeForce GTX 1650 Ti GPU.

Method	Diversity	nfcorpus		scifact		robust		covid		news	
		LLM	Total	LLM	Total	LLM	Total	LLM	Total	LLM	Total
Dense	n/a	0.0	0.1	0.0	0.1	0.0	0.4	0.0	0.2	0.0	0.4
Dense (QR)	n/a	0.9	0.9	0.8	0.9	1.1	1.4	0.8	1.0	0.6	1.1
PW (QR)	n/a	5.0	5.0	5.0	5.0	5.6	6.0	4.9	5.0	5.5	5.9
LW-one-call (QR)	n/a	4.1	4.1	4.3	4.3	4.4	4.8	2.7	2.8	4.5	5.0
LW-window (QR)	n/a	8.3	8.3	9.7	9.8	11.9	12.3	7.3	7.4	15.7	16.7
ReBOL-QR-Rand.	Top-B	6.0	6.1	7.1	7.3	5.7	5.9	5.2	5.6	5.7	6.0
ReBOL-QR-Top-k	n/a	5.7	5.8	6.3	6.4	6.0	6.2	5.9	6.0	5.7	5.9
ReBOL-QR-Greedy	Top-B	5.9	6.2	5.8	6.1	6.2	6.7	5.4	5.9	5.7	6.3
ReBOL-QR-UCB	Top-B	6.0	6.3	7.2	7.6	5.9	8.2	5.4	6.4	5.8	8.4
ReBOL-QR-UCB	KB	5.1	6.1	4.8	5.9	8.1	29.2	7.5	14.7	7.5	31.1
ReBOL-QR-Greedy	MMR	5.4	5.9	5.9	6.6	6.4	13.3	6.4	8.8	6.7	14.3
ReBOL-QR-UCB	MMR	5.4	5.9	5.1	5.7	5.9	14.4	7.1	10.0	8.6	18.4

effect of using different encoder models. Another direction for future work is improving scalability for very large corpora, possibly via clustering.

Several broader risks also arise when using LLMs for large-scale ranking and relevance assessment. First, LLMs may reproduce or amplify societal biases learned during pretraining, which can lead to harmful or unfair retrieval outcomes. Second, LLM-based systems remain vulnerable to adversarial prompt manipulation (e.g., prompt injection or "jailbreaking"), which can compromise system safety or reliability. Third, LLM relevance judgments themselves may be incorrect or inconsistent, particularly in ambiguous or domain-specific cases. Such errors may propagate through retrieval pipelines and could be problematic in high-stakes applications where ranking quality is critical.

As per ARR guidelines, we disclose the use of AI assistants for assistance with coding and grammar.

Acknowledgments

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grants funded by the Korea government (MSIT) (RS-2022-00143911, AI Excellence Global Innovative Leader Education Program; RS-2024-00457882, National AI Research Lab Project).

References

Jaime Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336.

Kaustubh D Dhole and Eugene Agichtein. 2024. Genqrensemble: Zero-shot llm ensemble prompting for generative query reformulation. In *European Conference on Information Retrieval*, pages 326–335. Springer.

Kaustubh D Dhole, Ramraj Chandradevan, and Eugene Agichtein. 2024. Generative query reformulation using ensemble prompting, document fusion, and relevance feedback. *arXiv preprint arXiv:2405.17658*.

Luyu Gao and Jamie Callan. 2021. Condenser: a pre-training architecture for dense retrieval. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 981–993.

Roman Garnett. 2023. *Bayesian optimization*. Cambridge University Press.

David Ginsbourger, Rodolphe Le Riche, and Laurent Carraro. 2010. Kriging is well-suited to parallelize optimization. In *Computational intelligence in expensive optimization problems*, pages 131–162. Springer.

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

Rolf Jagerman, Honglei Zhuang, Zhen Qin, Xuanhui Wang, and Michael Bendersky. 2023. Query expansion by prompting large language models. *arXiv preprint arXiv:2305.03653*.

Omar Khattab and Matei Zaharia. 2020. 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, pages 39–48.

Junyoung Kim, Anton Korikov, Jiazhou Liang, Justin Cui, Yifan Simon Liu, Qianfeng Wen, Mark Zhao, and Scott Sanner. 2026. Bayesian active learning with gaussian processes guided by llm relevance scoring for dense passage retrieval. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 9884–9898. Association for Computational Linguistics.

Anton Korikov, Pan Du, Scott Sanner, and Navid Rekabsaz. 2025. Batched self-consistency improves llm relevance assessment and ranking. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 32675–32691.

Anton Korikov, George Saad, Ethan Baron, Mustafa Khan, Manav Shah, and Scott Sanner. 2024. Multi-aspect reviewed-item retrieval via llm query decomposition and aspect fusion. In *Proceedings of the SIGIR 2024 Workshop on Information Retrieval's Role in RAG Systems*, Washington, DC, USA. CEUR Workshop Proceedings. SIGIR 2024 Workshop on Information Retrieval's Role in RAG Systems.

Ivica Kostric and Krisztian Balog. 2024. A surprisingly simple yet effective multi-query rewriting method for conversational passage retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2271–2275.

Sheng-Chieh Lin, Jheng-Hong Yang, Rodrigo Nogueira, Ming-Feng Tsai, Chuan-Ju Wang, and Jimmy Lin. 2021. Multi-stage conversational passage retrieval: An approach to fusing term importance estimation and neural query rewriting. *ACM Transactions on Information Systems (TOIS)*, 39(4):1–29.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Yifan Simon Liu, Qianfeng Wen, Jiazhou Liang, Mark Zhao, Justin Cui, Anton Korikov, Armin Toroghi, Junyoung Kim, and Scott Sanner. 2026. Multimodal item scoring for natural language recommendation via gaussian process regression with llm relevance judgments. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 36859–36876. Association for Computational Linguistics.

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*.

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, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, and 1 others. 2024. Large language models are effective text rankers with pairwise ranking prompting. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1504–1518.

Stephen Robertson and Hugo Zaragoza. 2009. *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc.

Devendra Sachan, Mike Lewis, Mandar Joshi, Armen Aghajanyan, Wen-tau Yih, Joelle Pineau, and Luke Zettlemoyer. 2022. Improving passage retrieval with zero-shot question generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3781–3797.

Gerard Salton, Anita Wong, and Chung-Shu Yang. 1975. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022. 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*, pages 3715–3734.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Is ChatGPT good at search? investigating large language models as re-ranking agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14918–14937.

Nandan Thakur, Luiz Bonifacio, Maik Fröbe, Alexander Bondarenko, Ehsan Kamalloo, Martin Potthast, Matthias Hagen, and Jimmy Lin. 2024. Systematic evaluation of neural retrieval models on the touché 2020 argument retrieval subset of beir. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1420–1430.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663*.

Petter Törnberg. 2024. Best practices for text annotation with large language models. *Sociologica*, 18(2):67–85.

Shivani Upadhyay, Ronak Pradeep, Nandan Thakur, Nick Craswell, and Jimmy Lin. 2024. Umbrela: Umbrela is the (open-source reproduction of the) BING relevance assessor. *arXiv preprint arXiv:2406.06519*.

Cornelis Joost Van Rijsbergen. 2004. *The geometry of information retrieval*. Cambridge University Press.

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

Qianfeng Wen, Yifan Liu, Justin Cui, Joshua Zhang, Anton Korikov, George-Kirollos Saad, and Scott Sanner. 2025. A simple but effective elaborative query reformulation approach for natural language recommendation. *arXiv preprint arXiv:2510.02656*.

A Prompts

Figure 3-4 show our prompts for query reformulation and listwise reranking.

```
Given an initial query, generate a set
of {{k_new}} new queries so that
together, this set of {{k_tot}} queries
(new plus old) could help find relevant
documents.
<inputs + format instructions>
```

Figure 3: Query Reformulation Prompt

```
Return the best ordering of the
documents given the query. In JSON
form, output a list of document ID
strings with a list length of exactly k
ids.
<inputs + format instructions>
```

Figure 4: Listwise reranking prompt — real document IDs are simplified before prompting to ["d1" ,..., "dk"] for k documents.

B Ablations

Table 6: LLM relevance observation model ablation with GPT-5.2 and a cross-encoder (ms-marco-MiniLM-L6-v2) showing Recall(R)@100 and NDCG(N)@10 for ReBOL (Top-B), with **bold** indicating the best overall method and underline the best absolute relevance scorer. In general, ReBOL continues to display strong performance though it is relatively weaker with the cross-encoder.

Method	GPT-5.2				Cross-Encoder			
	covid		news		covid		news	
	R@100	N@10	R@100	N@10	R@100	N@10	R@100	N@10
Dense (IQ)	9.5	51.8	42.3	38.2	9.5	51.8	42.3	38.2
PW (IQ)	9.5	73.8	42.3	45.9	9.5	**74.0**	42.3	45.6
LW-window (IQ)	9.5	78.0	42.3	52.4	n/a	n/a	n/a	n/a
Dense (QR)	8.0	45.4	45.7	43.3	8.0	45.4	45.7	43.3
PW (QR)	8.0	63.5	45.7	47.5	8.0	67.4	45.7	46.5
LW-window (QR)	8.0	66.7	45.7	**53.3**	n/a	n/a	n/a	n/a
ReBOL-IQ-Greedy	15.0	79.7	**55.0**	45.9	12.7	70.4	47.0	47.4
ReBOL-IQ-UCB	14.9	76.8	54.2	47.5	13.0	70.2	47.8	47.1
ReBOL-QR-Greedy	15.2	**80.8**	54.0	50.8	**13.5**	73.8	47.2	**49.3**
ReBOL-QR-UCB	**15.4**	80.2	53.1	47.9	12.4	65.8	**48.6**	49.2

Table 7: Ablation study with $k = 50$ and 50 observations for ReBOL (Top B)

Method	nfcorpus		scifact		robust		covid		news	
	R@50	N@10	R@50	N@10	R@50	N@10	R@50	N@10	R@50	N@10
Dense (IQ)	25.6	31.6	87.9	64.9	25.7	40.5	5.6	51.8	33.8	38.2
PW (IQ)	25.6	37.2	87.9	70.3	25.7	57.2	5.6	70.6	33.8	47.4
LW-one-call (IQ)	25.6	35.7	87.9	72.1	25.7	52.2	5.6	70.3	33.8	46.7
LW-window (IQ)	25.6	37.1	87.9	72.8	25.7	55.7	5.6	75.1	33.8	49.1
Dense (QR)	28.0	33.7	92.9	70.3	28.0	44.2	5.0	45.4	36.8	43.3
PW (QR)	28.0	39.3	92.9	72.3	28.0	59.6	5.0	62.3	36.8	46.4
LW-one-call (QR)	28.2	39.1	92.9	75.7	28.0	54.2	5.0	62.7	36.8	47.1
LW-window (QR)	28.2	40.2	92.9	**76.6**	28.0	57.6	5.0	64.5	36.8	48.1
ReBOL-Rand.(IQ)	23.8	24.9	87.5	57.7	26.0	39.1	5.7	47.1	33.6	37.8
ReBOL-Top-k (IQ)	29.4	38.3	92.1	66.7	32.5	59.5	8.1	**75.9**	38.1	45.9
ReBOL-Greedy (IQ)	28.7	38.9	91.5	70.2	34.1	59.1	8.2	73.4	37.7	47.5
ReBOL-UCB (IQ)	29.2	37.9	90.9	69.2	33.6	58.3	**8.6**	72.4	39.7	45.5
ReBOL-Rand. (QR)	26.7	32.6	92.6	70.4	29.4	47.1	6.8	56.8	38.1	45.3
ReBOL-Top-k (QR)	29.9	39.6	93.0	73.3	32.3	59.6	7.7	73.0	34.1	45.8
ReBOL-Greedy (QR)	30.3	40.0	**94.0**	73.1	35.1	**62.9**	**8.6**	75.4	37.0	44.3
ReBOL-UCB (QR)	**31.9**	**40.9**	93.7	73.9	**35.4**	60.8	8.4	74.2	**40.6**	**49.2**

Table 8: Ablation study with $k = 200$ and 200 observations for ReBOL (Top B)

Method	nfcorpus		scifact		robust		covid		news	
	R@100	N@10	R@100	N@10	R@100	N@10	R@100	N@10	R@100	N@10
Dense (IQ)	30.8	31.6	92.8	64.9	32.7	40.5	9.5	51.8	42.3	38.2
PW (IQ)	32.3	38.3	95.2	69.9	38.0	61.0	11.6	72.0	46.7	44.8
LW-one-call (IQ)	30.7	33.6	93.1	69.2	33.3	48.7	10.1	72.8	42.3	41.4
LW-window (IQ)	32.3	37.3	95.2	76.0	35.8	58.5	10.4	**81.5**	44.6	48.3
Dense (QR)	33.3	34.7	95.8	71.0	35.0	44.4	8.3	46.3	46.1	42.2
PW (QR)	34.9	38.5	97.0	68.3	40.5	60.8	10.3	66.3	49.8	43.5
LW-one-call (QR)	33.5	36.2	95.5	72.2	35.1	49.8	9.1	70.6	45.8	43.5
LW-window (QR)	34.8	40.5	97.0	**77.1**	38.1	59.3	9.1	77.4	48.4	**50.0**
ReBOL-Rand (IQ).	30.8	25.3	90.7	49.5	31.9	38.2	7.3	39.5	40.8	36.1
ReBOL-Top-k (IQ)	36.9	39.1	95.4	66.5	40.8	60.4	14.0	77.5	46.8	43.0
ReBOL-Greedy (IQ)	38.0	39.1	96.3	69.3	49.5	61.6	16.8	75.0	53.8	45.0
ReBOL-UCB (IQ)	37.8	37.9	95.0	68.5	49.1	61.5	16.3	75.9	**55.8**	40.4
ReBOL-Rand. (QR)	34.3	33.4	95.8	69.5	36.1	47.2	12.6	62.3	46.7	43.8
ReBOL-Top-k (QR)	36.3	40.2	96.5	73.9	41.1	61.0	13.4	74.6	46.6	44.8
ReBOL-Greedy (QR)	38.6	40.1	**98.7**	74.1	**50.3**	63.5	16.8	75.8	53.3	45.7
ReBOL-UCB (QR)	**39.2**	**40.7**	96.0	73.6	50.0	**64.3**	**17.2**	77.0	54.8	47.1