Col-Bandit: Query-Time Top-K Estimation for Late-Interaction Retrieval

Anonymous ACL submission

Abstract

Multi-vector late-interaction retrievers such as ColBERT achieve state-of-the-art quality, but their query-time cost is dominated by exhaustively computing token-level MaxSim interactions for every candidate document. The MaxSim scores of N candidates against T query tokens form an $N \times T$ matrix whose row-sums are the late-interaction scores, and identifying the top-K rarely requires every entry. We introduce Col-Bandit,[1] a query-time estimator of the exhaustive-MaxSim top-K: it reveals matrix entries in batches, maintains a finite-population Bernstein–Serfling confidence interval on each candidate's score, and permanently drops any document whose upper bound falls below the K-th largest lower bound, computing only the cells needed to separate the top-K. A single relaxation knob $\alpha_{\mathrm{ef}} \in (0, 1]$ tunes the compute–fidelity trade-off. We deploy $\alpha_{\mathrm{ef}}{=}0.2$, while $\alpha_{\mathrm{ef}}{=}1$ admits a δ-PAC guarantee under a simplified radius. On BEIR and REAL-MM-RAG, Col-Bandit preserves $\geq$ 90% fidelity to the exhaustive top-5 on every corpus while cutting MaxSim FLOPs by up to $\sim$8$\times$, for up to $\sim$13$\times$ single-thread CPU speedups across x86 and ARM. A drop-in reranking layer, it needs no retraining or index changes.

1 Introduction

Multi-vector late-interaction retrievers, such as ColBERT (Khattab and Zaharia, 2020), have emerged as a powerful alternative to single-vector dense retrieval. By representing each query and document as a *set* of token embeddings, these models capture fine-grained semantic matches that single-vector representations miss (Wang et al., 2023; Formal et al., 2021). This paradigm has been widely adopted in recent text and multimodal systems (Faysse et al., 2024; Team, 2025a; Warner et al., 2025; Team, 2025b; Xu et al., 2025; Günther et al., 2025), becoming a standard foundation for high-accuracy neural retrieval. However, this granularity comes with a cost. Unlike single-vector retrieval, where scoring is a cheap dot product, exact late interaction requires evaluating a grid of token-level operations (MaxSim) for every document. Consequently, this computation often becomes the bottleneck in modern pipelines, motivating methods that reduce these operations without sacrificing ranking fidelity (Santhanam et al., 2022a; Engels et al., 2023).

The "Hiring" Analogy. Consider a manager hiring the top-K candidates from N applicants, where each takes T independent tests and the final score is the sum. An efficient manager proceeds in rounds: each round administers a small batch of new tests to the surviving applicants, then drops those who cannot reach the top-K even if they ace every untaken test. Standard late-interaction retrieval skips the rounds: it scores every token interaction on every document, even those already ruled out.

Our Approach: Col-Bandit (CB). We view this as *progressive matrix completion*: the token-level scores are values in a table revealed on demand, and our objective is to estimate the exhaustive scorer's Top-K identity from a partially revealed matrix, minimizing computation while maintaining a user-defined level of statistical reliability (Figure 2). To this end, we introduce Col-Bandit (CB), a *purely query-time* algorithm that operates directly on *vanilla ColBERT*. Col-Bandit exploits a fundamental asymmetry that index-time methods cannot use: **the query is only known at query time**, so only at query time can the system decide which document tokens matter for *this* query, and the relevant cells of the MaxSim matrix are unknowable a priori. We refer to the algorithm as CB and to its concrete kernel implementation built on the NUMKONG (Vardanian) C extension

[1] Code will be released as open-source software upon publication.

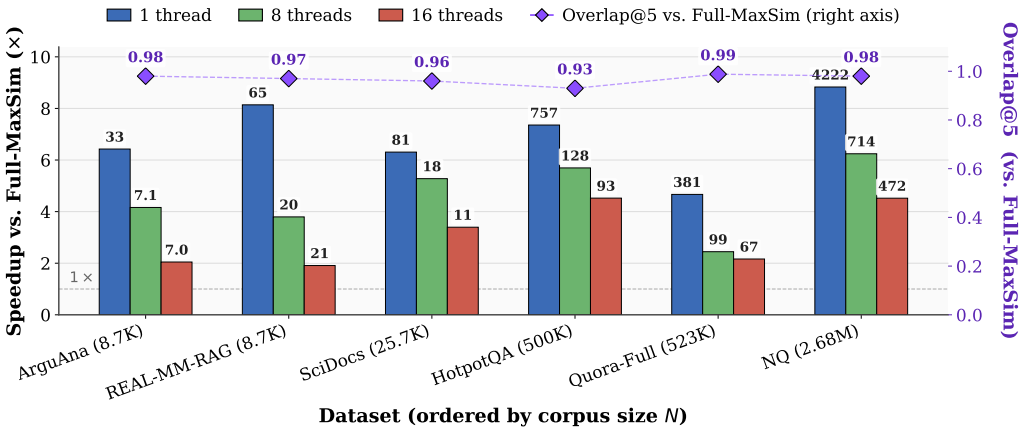

Figure 1: **Wall-clock speedup of Col-Bandit vs. Full-MaxSim across BEIR and REAL-MM-RAG** on CPU-S at the deployed knob (α_{ef}=0.2, M=5, δ=0.01, K=5). Bars are speedup factors at 1/8/16 threads. Numbers above each bar are Col-Bandit per-query latency in ms. Datasets are ordered by corpus size N. Right-axis purple diamonds: Overlap@5 vs. Full-MaxSim's exhaustive top-K. Fidelity stays ≥ 0.93 on CPU-S for every corpus (HotpotQA-500 K is the floor). See Table 3 for the cross-platform breakdown. REAL-MM-RAG is the four multimodal corpora merged into one collection. Extended sweeps at other α_{ef} and K values in Appendix B.3.

as CB-NK. Unlike prior acceleration methods that *lossily* compress or distill document representations, Col-Bandit targets the exact-MaxSim top-K (surviving candidates are always rescored exactly on all T tokens, and the rule that eliminates the rest is provably correct at α_{ef}=1) and requires **no retraining and no changes to the model or retrieval index**. Its fast kernel uses only a one-time offline repack of the candidate embeddings into a cache-friendly layout (Appendix D). It composes with retrieval-side systems such as PLAID and MUVERA rather than competing with them. We formulate the task as a *finite-population Top-K identification problem*. By exploiting the fact that document token sequences are finite, we utilize the empirical Bernstein–Serfling inequality (Bardenet and Maillard, 2015) to construct tighter confidence intervals than standard bandit approaches.

Contributions.

- **Formulation.** We cast late-interaction reranking as a finite-population Top-K identification problem using a progressive scoring framework, exploiting the query at query time to decide which MaxSim cells to compute, a signal that index-time accelerators cannot use.
- **Algorithm.** We introduce Col-Bandit, a progressive multi-round elimination algorithm that leverages the empirical Bernstein–Serfling inequality (Bardenet and Maillard, 2015) for tight per-document confidence bounds, with a tunable relaxation parameter α_{ef} that provides a δ-PAC certificate at α_{ef}=1 (under the simplified radius of Eq. 7, see Remark 2). The deployed default α_{ef}=0.2 is a calibrated relaxation that retains the Pareto-dominant cost–fidelity profile we report in §5.
- **Drop-in Acceleration.** We demonstrate a **7.0×/4.6×** (1/8 thread) wall-clock speedup on a server CPU (AMD EPYC 7763, AVX2) and **12.9×/10.3×** (1/8 thread, K=5) on Apple M1 Max (NEON, four-corpus subset), across BEIR and REAL-MM-RAG (nine corpora, up to 2.68 M documents, Figure 1), preserving Overlap@5 $\geq$ 0.90 on every corpus tested ($\geq$ 0.96 on CPU-S for every corpus but HotpotQA-500 K), with no index modifications or retraining. Latency scales *sub-linearly* in K.

2 Background and Related Work

2.1 Preliminaries: Late Interaction Retrieval

ColBERT Late Interaction Scoring. Consider a query Q and a document d from a collection $\mathcal{D}$ of size N. ColBERT represents both as sets of token embeddings:

$$Q = \{\mathbf{q}_t\}_{t=1}^{T}, \quad \mathbf{E}(d) = \{\mathbf{e}_{d,j}\}_{j=1}^{L_d},$$

where $\mathbf{q}_t, \mathbf{e}_{d,j} \in \mathbb{R}^l$, l is the embedding dimension, T the query length, and L_d the document length. The score is computed via *late interaction*: for each query token $t \in [T]$, ColBERT identifies the most

similar document token (MaxSim) and sums:

$$h(d,t) \triangleq \max_{j\in[L_d]} \mathrm{sim}(\mathbf{e}_{d,j}, \mathbf{q}_t), \tag{1}$$

$$S(d;Q) \triangleq \sum_{t=1}^{T} h(d,t), \tag{2}$$

where $\mathrm{sim}(\cdot,\cdot)$ is a similarity function (typically cosine).[2]

Top-K Ranking. The ranking objective is to identify the K documents from a search set $\mathcal{D}$ (the full corpus, or a candidate set produced by an upstream stage) with the highest MaxSim scores:

$$\mathcal{T}_K^\star \triangleq \arg\operatorname*{topK}_{d\in\mathcal{D}} S(d;Q). \tag{3}$$

Index-Time vs. Query-Time. Retrieval systems separate *index-time* (offline) representation/index construction from *query-time* (online) scoring. In late-interaction systems the latter is typically a *reranking* stage. **This separation matters for pruning:** index-time methods must commit before seeing the query and prune conservatively, while query-time methods can use the actual query to decide what to compute. Col-Bandit is, to our knowledge, the first method to exploit this signal at the atomic MaxSim-cell level. Standard reranking evaluates all $N \times T$ MaxSim values $h(d,t)$ (Eq. 2), which dominates query-time cost even after candidate retrieval.

2.2 Related Work

We categorize related work by *when* and *what* they prune (visual taxonomy in Appendix E). To our knowledge, Col-Bandit is the first method to *adaptively use the query* to prune at the *atomic* MaxSim-cell level during query-time scoring.

Index-time accelerators. Centroid compression and fixed-dimensional encodings, including PLAID (Santhanam et al., 2022a), ColBERTv2 (Santhanam et al., 2022b), MUVERA (Dhulipala et al., 2024), LEMUR (Jääsaari et al., 2026), and WARP (Scheerer et al., 2025), accelerate retrieval by quantizing or projecting document representations before the query arrives. Offline token pruning (Lassance et al., 2021; Tonellotto and Macdonald, 2021; Clavié et al., 2024; MacAvaney et al., 2025) reduces the index size N or query length T by permanently discarding low-utility tokens. Both families must commit at index time, before seeing the query. Col-Bandit is *orthogonal* and stacks on top, since any such system can produce the candidate set $\mathcal{D}$ that Col-Bandit reranks.

Bound-based skipping. In sparse retrieval, WAND (Broder et al., 2003) and BMW (Ding and Suel, 2011) use per-term upper bounds to skip low-scoring documents, and DESSERT (Engels et al., 2023) applies similar bound-based reasoning to approximate set search. Col-Bandit brings this idea to *dense late-interaction*, pruning atomic MaxSim cells $h(d,t)$ with statistical (rather than term-level) bounds.

Top-K arm identification. Col-Bandit extends fixed-confidence Top-K arm identification (Kalyanakrishnan et al., 2012; Chen et al., 2014; Audibert and Bubeck, 2010) from stochastic arms to a *finite population sampled without replacement* (each MaxSim row is a fixed set of T deterministic values). We therefore use the empirical Bernstein–Serfling concentration of Bardenet and Maillard (2015) in place of the standard sub-Gaussian radius (§4.1).

3 Problem Formulation

We construct Col-Bandit as a statistical estimator of $\mathcal{T}_K^\star$ defined in Eq. (3). Given a query Q, it outputs $\hat{\mathcal{T}}_K \subset \mathcal{D}$. Under confidence parameter δ and relaxation α_{ef}, it recovers $\mathcal{T}_K^\star$ with probability at least $1-\delta$ when $\alpha_{\mathrm{ef}} = 1$. We cast this as a fixed-confidence Multi-Armed Bandit (MAB) problem over the sparsely observed, finite-population MaxSim matrix.

3.1 The MaxSim Matrix and Observation Model

Consider a query Q with T tokens and a search set of N documents, $\mathcal{D} = \{d_1, \ldots, d_N\}$. We define the implicit *MaxSim Matrix*, $H \in \mathbb{R}^{N\times T}$, where each entry corresponds to the maximum similarity (Eq. 2) of a query token with a document's tokens:

$$H_{i,t} \triangleq h(d_i,t) = \max_{j\in[L_{d_i}]} \mathrm{sim}(\mathbf{e}_{d_i,j}, \mathbf{q}_t). \tag{4}$$

The total late-interaction score for document i is the row-sum:

$$S_i \triangleq \sum_{t=1}^{T} H_{i,t}. \tag{5}$$

[2] More generally, we assume sim is bounded in a known interval $[a,b]$ (e.g., $[-1,1]$ for cosine on normalized vectors), hence each $h(d,t)$ is also bounded.

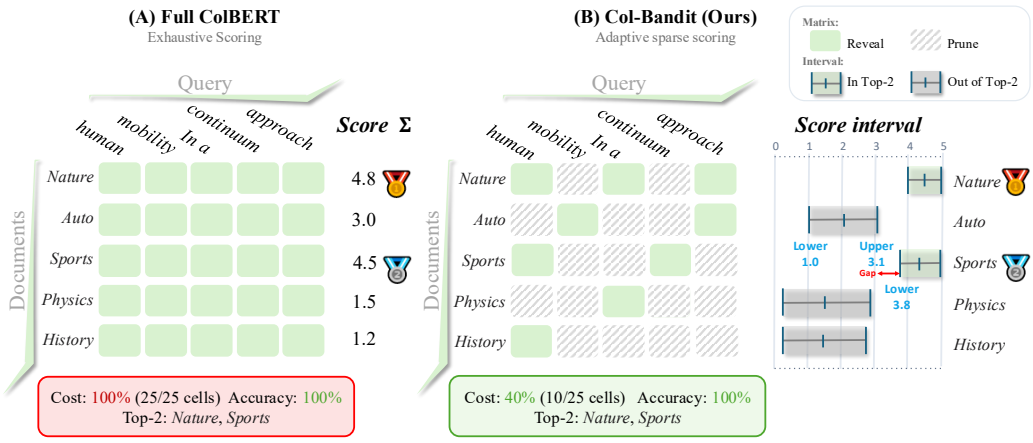

Figure 2: **Intuition: the top-K is identifiable from a partial view of the MaxSim matrix H.** Given a query and a candidate set, the goal is to identify the top-K documents under exhaustive MaxSim. **(A)** Full ColBERT scores every cell of the $N \times T$ matrix. **(B)** Col-Bandit reveals only a subset of cells (green) and skips the rest (hatched), maintaining a confidence interval $[\mathrm{LCB}_i, \mathrm{UCB}_i]$ on each document's total score. Once a positive *separation gap* opens between the weakest winner's LCB and the strongest loser's UCB, the top-K is determined at a fraction of the compute. This panel illustrates *why* partial observation suffices. The deployed reveal schedule (shared random token batches with whole-document pruning) is given in Algorithm 1.

Our objective is to identify the set of indices $\mathcal{T}_K^\star$ corresponding to the K documents with the highest scores S_i.

At any step, the algorithm holds an observed set $\Omega \subseteq [N] \times [T]$ with per-document indices $\mathcal{O}_i \triangleq \{t : (i,t) \in \Omega\}$ and $\mathcal{U}_i \triangleq [T] \setminus \mathcal{O}_i$. Revealing $(i,t) \notin \Omega$ incurs unit cost and returns $H_{i,t}$. We measure computational cost via *coverage*, defined as the fraction of the matrix revealed. At any time step of our algorithm the cost is:

$$\gamma(\Omega) \triangleq \frac{|\Omega|}{N \times T} = \frac{1}{NT} \sum_{i=1}^{N} |\mathcal{O}_i|. \tag{6}$$

4 Method: Col-Bandit

Overview. Figure 2 previews Col-Bandit: from a partially revealed MaxSim matrix we maintain per-document lower/upper bounds (§4.1) and identify the top-K once the weakest winner's lower bound exceeds the strongest loser's upper bound (a positive *separation gap*), and §4.2 gives the round-by-round procedure that drives this gap.

4.1 Decision Bounds

Let $n_i = |\mathcal{O}_i|$ be the number of revealed query-token positions (cells) for document i, $\widehat{\mu}_i = \frac{1}{n_i} \sum_{t \in \mathcal{O}_i} H_{i,t}$ the empirical mean, and $\widehat{S}_i \triangleq T\widehat{\mu}_i$ the partial-sum estimator used to order candidates. Using the known global support $[a,b]$ of unrevealed entries (*e.g.*, $[-1,1]$ for cosine similarity), the deterministic hard bounds are $LB_i^{\mathrm{hard}} = \sum_{t \in \mathcal{O}_i} H_{i,t} + (T - n_i)\, a$ and $UB_i^{\mathrm{hard}} = \sum_{t \in \mathcal{O}_i} H_{i,t} + (T - n_i)\, b$. Tighter per-cell bounds (*e.g.*, per-document token-norm or centroid-based upper/lower bounds) would tighten both LB_i^{hard} and UB_i^{hard}. We leave them for future work and use the simpler global support throughout. Correctness depends only on $H_{i,t} \in [a,b]$ holding. If sim is not strictly bounded (*e.g.*, un-normalized embeddings), widening $[a,b]$ keeps the bounds valid and only loosens the radii, never breaking elimination safety. We combine these with an empirical Bernstein–Serfling style decision radius (Bardenet and Maillard, 2015),

$$r_i^{\mathrm{eff}} \triangleq \alpha_{\mathrm{ef}} \cdot T\widehat{\sigma}_i \sqrt{\frac{2\log(cNT/\delta)}{n_i}} \cdot \sqrt{\rho_{n_i}}, \tag{7}$$

where $\widehat{\sigma}_i$ is the empirical standard deviation over revealed entries and ρ_{n_i} is a finite-population correction with $\rho_{n_i} \to 0$ as $n_i \to T$ (Appendix A). The relaxation $\alpha_{\mathrm{ef}} \in (0,1]$ controls conservativeness: $\alpha_{\mathrm{ef}}{=}1$ recovers the unshrunk empirical Bernstein–Serfling form, while $\alpha_{\mathrm{ef}}{<}1$ tightens the radius. The hybrid decision interval is

$$\begin{aligned} \mathrm{LCB}_i &= \max(LB_i^{\mathrm{hard}},\ \widehat{S}_i - r_i^{\mathrm{eff}}), \\ \mathrm{UCB}_i &= \min(UB_i^{\mathrm{hard}},\ \widehat{S}_i + r_i^{\mathrm{eff}}). \end{aligned} \tag{8}$$

4.2 Batched Progressive Elimination

From rule to schedule. A per-cell loop would be dominated by dispatch overhead (a MaxSim cell is only a few FLOPs), so we run Col-Bandit in

Algorithm 1 Col-Bandit: batched progressive elimination for top-K.

Require: Query Q (T tokens), candidates $\mathcal{D}$ (N docs); $K, M, B, \alpha_{\text{ef}}, \delta$.
Ensure: Estimated top-K set $\widehat{\mathcal{T}}_K$.
1: $r \leftarrow 1$; $\mathcal{A}_1 \leftarrow [N]$; $\Omega \leftarrow \emptyset$; draw a uniform permutation π of $[T]$.
2: **while** $|\mathcal{A}_r| > K + M$ **and** $(r-1)B < T$ **do**
3: Reveal the next B cells of π for every $i \in \mathcal{A}_r$ (one SIMD pass); add them to Ω and refresh $[\mathrm{LCB}_i, \mathrm{UCB}_i]$ via Eq. 8.
4: $\tau_r \leftarrow K$-th largest LCB_i over $i \in \mathcal{A}_r$.
5: $\mathcal{A}_{r+1} \leftarrow \{i \in \mathcal{A}_r : \mathrm{UCB}_i \geq \tau_r\}$; $r \leftarrow r + 1$.
6: **end while**
7: Rescore each survivor: $S_i \leftarrow \sum_{t=1}^{T} H_{i,t}$ via the fused MaxSim kernel.
8: **return** $\widehat{\mathcal{T}}_K \leftarrow \arg\mathrm{topK}_{i \in \mathcal{A}_r} S_i$.

rounds over an *active set* $\mathcal{A}_r \subseteq [N]$ ($\mathcal{A}_1 = [N]$). Each round reveals the next B cells for every $i \in \mathcal{A}_r$ in one vectorized pass, refreshes the bounds, and eliminates any document whose UCB_i falls below the K-th largest LCB over $\mathcal{A}_r$. Per-round bookkeeping amortizes across $B|\mathcal{A}_r|$ cells.

Pay-to-prune. Each round costs $B|\mathcal{A}_r|$ evaluations, but every eliminated document saves its remaining $T - rB$ cells, so the active set shrinks monotonically and the cumulative budget is $\sum_r B|\mathcal{A}_r| \ll NT$ in practice (§5).

We instantiate this as Algorithm 1 (fully annotated in Appendix D.1). The reveal schedule is a single per-query permutation π of $[T]$, drawn uniformly with a fixed RNG seed and shared across surviving documents (the uniform-without-replacement structure is what makes Theorem 1 applicable, see Appendix C). A safety margin M (M=5 by default) preserves up to $K+M$ borderline survivors, which are then rescored on all T query tokens via the same fused MaxSim kernel as Full-MaxSim, so the final survivor scores are *bit-identical* to the exhaustive baseline. Ties at the K-th boundary are broken by document index and deferred to the exact rescore. We use B=4 throughout, matching the SIMD register tile of our fused C kernel (Appendix D). The Bernstein–Serfling prefactor c in Eq. 7 and the $n_i \leq 1$ case are detailed in Appendix A.

Certified corner and deployed relaxation. Under uniform-without-replacement reveals, α_{ef}=1 admits a δ-PAC guarantee (under the simplified radius of Eq. 7, see Remark 2): $\mathbb{P}(\widehat{\mathcal{T}}_K = \mathcal{T}_K^\star) \geq 1-\delta$ (Theorem 1 and proof in Appendix C). *Sketch:* the empirical Bernstein–Serfling interval holds per row at a fixed reveal count. Setting the per-cell budget to $\delta/(NT)$ and union-bounding over all N documents and T reveal counts (the $\log(cNT/\delta)$ term in Eq. 7) makes every $[\mathrm{LCB}_i, \mathrm{UCB}_i]$ valid *simultaneously*, hence also at the data-dependent stopping time, side-stepping optional-stopping issues. A document is eliminated only once its UCB_i provably falls below the K-th largest LCB, so no true top-K member is dropped on this $1-\delta$ event. The constant c is the Bernstein–Serfling pre-factor, set to 1 and absorbed into α_{ef} (Appendix A). For α_{ef}<1 the radii shrink and elimination is more aggressive, trading the certificate for coverage. The deployed default α_{ef}=0.2 is this calibrated relaxation, with no formal certificate but ≥ 0.90 overlap with the exhaustive top-K on every corpus tested (Overlap@K defined in §5).

5 Experiments

5.1 Experimental Setup

Corpora. Five BEIR language corpora (ArguAna, SciDocs, NQ-2.68M, HotpotQA-500K, Quora-Full (Thakur et al., 2021)) and four REAL-MM-RAG (Wasserman et al., 2025) multimodal corpora (FinSlides, FinReport, TechSlides, TechReport), plus a merged MM set (the four REAL-MM-RAG corpora pooled) for the CPU wall-clock benchmarks (CPU-S and CPU-M1). Per-corpus details in Appendix A.2 (Table 4). N is the full corpus rerank size (e.g., N=2.68M for NQ).
Encoders. ColBERTv2 (Santhanam et al., 2022b) on all text corpora, Jina-ColBERTv2 (Jha et al., 2024) ($d \in \{128, 64\}$ Matryoshka) on four BEIR corpora, and Granite Vision Embedding 3.2 (Team, 2025a) for REAL-MM-RAG.
Hardware. CPU-S = AMD EPYC 7763 (AVX2 server), CPU-M1 = Apple M1 Max (NEON laptop), and GPU = NVIDIA A100 80 GB. A single NUMKONG (Vardanian) C extension drives both CPUs via SIMD-backend swap (Appendix D.2).
Baselines. Full-MaxSim (exhaustive oracle), Doc-Uniform (non-adaptive random-cell reveal at coverage γ, Appendix A.3), and Ball carving (Dhulipala et al., 2024) (Stage-2 query-time pruning peer, Appendix A.3). All methods are built on the same NUMKONG kernel and differ in what they reveal or compress per query (Appendix A.3).
Metrics. Our metric measures how Col-Bandit reproduces Full-MaxSim's top-K set:

$$\text{Overlap@}K \triangleq \frac{|\mathcal{T}_K^\star \cap \hat{\mathcal{T}}_K|}{K}, \qquad (9)$$

Table 1: **Universal Efficiency Analysis: Overlap@K and nDCG@K at $K \in \{5, 50\}$.** Each cell: **mean coverage % (std across corpora)** to recover **95% (near-lossless)** of Full-MaxSim's own Overlap@K / nDCG@K per dataset (lower is better). The 90% threshold and per-dataset breakdowns are in Appendix B.2. Quora's nDCG/Recall/MRR are undefined under our harness (a qrels/doc-id mapping issue; Table 4) and do not contribute to the nDCG averages here, which match the Quora-excluded per-corpus means (Appendix B.2); Overlap@K is unaffected.

Method	Overlap@5	Overlap@50	nDCG@5	nDCG@50
ColBERTv2 (BEIR)				
Doc-Uniform	98% (3.8)	100% (0.0)	57% (10.3)	38% (11.7)
Ball-carving	46% (13.3)	43% (12.2)	29% (11.1)	23% (4.3)
Col-Bandit (Ours)	**14% (1.7)**	**18% (5.1)**	**13% (0.8)**	**13% (0.6)**
Jina-ColBERTv2 (BEIR)				
Doc-Uniform	100% (0.0)	100% (0.0)	59% (30.7)	57% (27.7)
Ball-carving	69% (32.6)	88% (21.3)	20% (6.3)	19% (5.6)
Col-Bandit (Ours)	**26% (16.9)**	**39% (17.8)**	**9% (3.5)**	**14% (7.5)**
Granite-Vision-Embedding (REAL-MM-RAG)				
Doc-Uniform	100% (0.0)	98% (4.1)	54% (18.6)	49% (16.0)
Ball-carving	76% (1.7)	76% (1.7)	28% (9.8)	27% (10.9)
Col-Bandit (Ours)	**22% (1.9)**	**41% (3.1)**	**15% (5.7)**	**15% (2.7)**

where $\mathcal{T}_K^\star$ is the exhaustive top-K and $\hat{\mathcal{T}}_K$ is Col-Bandit's estimate, with $\mathrm{Overlap@}K = 1$ iff the two sets coincide. We report $K \in \{5, 50, 100\}$, coverage γ (Eq. 6), and per-query wall-clock latency. Task-level metrics (nDCG, Recall, MRR) are in Appendix B.4–B.5.

CB knobs. Deployed: α_{ef}=0.2, M=5 (rescore margin), δ=0.01. Round size B=4 matches the SIMD register tile of the fused C kernel. Pareto sweeps span $\alpha_{\text{ef}} \in (0, 1]$ (Appendix D.3). The α_{ef}=1 corner satisfies Theorem 1.

5.2 Main Results: Coverage Savings Translate to Wall-Clock Latency

Col-Bandit reaches the exhaustive late-interaction top-K at a small fraction of the MaxSim cost, portably across server (AVX2) and edge (ARM NEON) CPUs.

Coverage savings. Table 1 reports the mean coverage budget needed to recover 95% (near-lossless) of Full-MaxSim's own Overlap@K and nDCG@K at $K \in \{5, 50\}$ across the BEIR (language) and REAL-MM-RAG (multimodal) suites. Col-Bandit dominates both non-adaptive baselines on every dataset. Figure 3 confirms this for GVE/FinSlides: the α_{ef}-swept frontier (compute cost vs. Overlap@5) lies strictly below Doc-Uniform and Ball-carving. The deployed α_{ef}=0.2 recovers ≈98% Overlap@5 at 19% cost. **Headline:** at the 95% near-lossless threshold, Col-Bandit reaches Overlap@5 at **14%/26%/22%** coverage on ColBERTv2/Jina/GVE respectively (**4.6–7.0**× savings), versus 98–100% for Doc-Uniform and 46–76% for Ball-carving, a 2–5× gap that holds on every corpus and at K=50 as well. Doc-Uniform's near-total coverage makes it a redundancy *upper bound*, so Ball-carving is the substantive baseline. The savings widen for ranking quality: nDCG@5 recovers at just **9**% coverage on Jina (**10.9**×) and **13**% on ColBERTv2 (**7.6**×). Recall@5 and MRR@5 follow the same coverage-savings pattern (Col-Bandit reaches 95% retention at 13–26% coverage on every corpus, with per-corpus tables in Appendix B.2). The looser 90% threshold is in Appendix B.1.

Wall-clock translation. Coverage savings translate to wall-clock speedups via a first-order model: every method here shares the same fused C MaxSim kernel (NUMKONG, Appendix D.2) and differs mainly in which (d, t) cells it touches (Col-Bandit uses an int8 elimination pass and an exact fp32 rescore, bit-identical to Full-MaxSim), so per-query latency is dominated by kernel work. The bandit bookkeeping (elimination decisions + the $K+M$ rescore) stays below 4% of total runtime across the eight corpora profiled in Appendix B.6, giving the heuristic $\text{Speedup} \sim 100/\text{Cov}\%$. Per-query wall-clock latency on CPU-S is reported in Table 2 (visualized as Figure 1, with extended sweeps at other α_{ef} and K values in Appendix B.3), and on CPU-M1 in Table 3. At the deployed knob (α_{ef}=0.2, M=5, δ=0.01), Col-Bandit preserves Overlap@$K \geq 0.90$ on every corpus and delivers a **7.0**× mean single-thread speedup on CPU-S (K=5, falling to 5.3× at K=100) and **12.9**×/**10.3**× (1/8 thread, K=5)

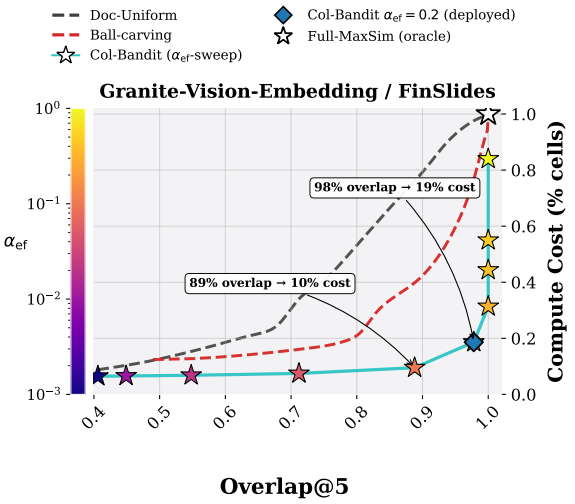

Figure 3: **Quality–coverage Pareto frontier (K=5), GVE/FinSlides.** Col-Bandit (sweeping α_{ef}) vs. Doc-Uniform and Ball-carving (MUVERA App. C.3). Col-Bandit dominates both baselines, reaching high fidelity at $\leq$ 20% of the cell budget. The deployed knob α_{ef}=0.2 marked, Full-MaxSim oracle at top-right. Additional comparisons in Appendix Figure 8.

on CPU-M1 (9.0×/7.0× at K=100). Against *maxsim-cpu* (Clavié and Lee, 2025), the published Rust SIMD MaxSim baseline, Col-Bandit is **4.8**× faster on average at 1 thread on CPU-M1 (K=5) at comparable fidelity (Table 3). Absolute end-task quality tracks the oracle: at ∼20% coverage Col-Bandit retains $\geq$ 98.7% of Full ColBERT's Recall@1/nDCG@1/MRR@1 (vs. 81–82% Ball-carving, 56% Doc-Uniform, Appendix B.4).

Commodity CPU vs. GPU at the deployed knob. Table 21 pairs 16-thread Col-Bandit on CPU-S against GPU PyTorch dense Full-MaxSim (K=5). This is an *architecture-level* pairing, not an apples-to-apples algorithm comparison. On the two text corpora with $N \geq$ 500 K (α_{ef}=0.2, Ov@5 $\geq$ 0.92), CPU-S lands within 1.06–1.41× GPU latency. NQ-2.68M overflows the 80 GB GPU budget (∼ 140 GB at fp16) yet CPU-S reranks it in 472 ms. On smaller corpora ($N \leq$ 26 K) GPU is 2–4× faster, as expected.

5.3 Sensitivity, Composability, and Calibration

Sub-linear scaling in K. Does the algorithm collapse outside the small-K rerank regime? Sweeping $K \in \{5, 100, 500, 1000\}$ at the deployed knob (Table 19), Col-Bandit latency grows far slower than K (sub-linearly) while Full-MaxSim is essentially flat: a 200× K-jump (5 → 1000) costs only 1.3× on HotpotQA and 2.7× on SciDocs, though 4.5× on the small ArguAna corpus. This reflects the adaptivity of the elimination phase rather than a property of K per se: with the K-margin rescore fixed at M=5, larger K only forces the elimination phase to retain a bigger active set at termination, which it absorbs cheaply on large corpora (per-document decisions amortize over many documents) but less so on small ones like ArguAna. The headline-speedup chart at K=100 is shown in Appendix B.3 (Figure 7).

Rescore margin M. We sweep $M \in \{0, 5, 10\}$ on SciDocs and HotpotQA-500 K at K=5, α_{ef}=0.2, δ=0.01, 1 thread, 50 queries per corpus. Without the rescore (M=0), borderline-eliminated documents leak through and Ov@5 drops to 0.93–0.94. Enabling M=5 lifts Ov@5 to $\geq$ 0.98 at a small latency cost (+6%/+21% on HotpotQA/SciDocs). M=10 matches M=5 on fidelity: five slots already absorb the eliminator's borderline mistakes (Appendix D.3, Table 25).

Cross-encoder: trends are not encoder-specific. Are the savings an artifact of ColBERTv2's embedding geometry? We re-run the universal-efficiency analysis with Jina-ColBERTv2 (Jha et al., 2024) (L_d≤256, d=128) and its d=64 Matryoshka projection on four BEIR corpora (Table 1, middle block, where Jina-NQ embeddings were not built and are marked "n/a" in Appendix B.2). At 95% Overlap@5, Col-Bandit costs **26**% coverage on Jina (vs. 100% Doc-Uniform, 69% Ball-carving). On nDCG@5 the saving widens to **9**% coverage (**10.9**×). The same conclusion holds at d=64, confirming the savings behave as a property of multi-vector representations, not of a single encoder.

Compression composability. Col-Bandit composes orthogonally with two complementary axes: *dimension reduction* (Matryoshka) and *token-axis pooling* (Ward (Clavié et al., 2024)). Under Matryoshka d=128 → 64, Ov@5 retention stays $\geq$ 0.95 on every Jina-ColBERTv2 BEIR corpus and coverage is essentially invariant to the projection (Appendix B.10). Under Ward k× token pooling ($k \in \{2, 4, 8\}$, per-doc proportional), the end-to-end Ov@5 loss decomposes empirically into a pooling-induced term (9–41 pp, median 20) and Col-Bandit's elimination cost (0.8–4.0 pp, median 2.0). Pooling dominates by an order of magnitude across the six (corpus, k) cells we measured (Appendix B.11, single-seed measurement). The two

Table 2: **Wall-clock CPU benchmarks on CPU-S** (AMD EPYC 7763, AVX2) at the deployed knob (α_{ef}=0.2, M=5, δ=0.01), **single thread**. Each cell shows K=5 / K=100. Ov@K is overlap with Full-MaxSim's exhaustive top-K (Eq. 9); Cov is the fraction of (d, t) MaxSim cells revealed. Col-Bandit delivers a **7.0× / 5.3×** (K=5 / K=100) mean single-thread speedup across the six CPU-S settings (five BEIR corpora and the REAL-MM-RAG aggregate) while preserving Ov@5 ≥ 0.93 on every corpus. Full-MaxSim latency is K-invariant up to measurement noise. Threads parallelise across queries, so multi-thread figures are throughput (per-query latency under concurrent load), not single-query speedup. Multi-thread (1t / 8t) breakdown in Appendix Table 17.

Dataset	N	**Ov@K**	**Cov**	**Full (ms)**	**CB (ms)**	**Sp.**
ArguAna	8.7 K	0.98 / 0.97	14% / 20%	211 / 210	**33 / 50**	6.4× / 4.2×
SciDocs	25.7 K	0.96 / 0.93	14% / 18%	513 / 515	**81 / 108**	6.3× / 4.8×
HotpotQA	500 K	0.93 / 0.93	13% / 14%	5,568 / 5,153	**757 / 825**	7.4× / 6.2×
Quora	522.9 K	0.99 / 0.97	13% / 13%	1,778 / 1,806	**381 / 393**	4.7× / 4.6×
NQ	2.68 M	0.98 / 0.97	13% / 13%	37,283 / 37,224	**4,222 / 4,384**	8.8× / 8.5×
MM	8.6 K	0.97 / 0.95	21% / 47%	527 / 523	**65 / 164**	8.1× / 3.2×
Mean	—	—	—	—	—	**7.0× / 5.3×**

Table 3: **Cross-platform benchmark on CPU-M1** (Apple M1 Max, NEON laptop) at the deployed knob, **single thread**. Each cell shows K=5 / K=100; ColBERTv2 on text, GVE on MM. Col-Bandit delivers a **12.9× / 9.0×** (K=5 / K=100) mean single-thread speedup vs. Full-MaxSim and **4.8× / 3.5×** vs. *m-cpu* (Clavié and Lee, 2025) (Rust SIMD baseline). Overlap@K can differ slightly from CPU-S because int8 dot products round differently across SIMD back-ends (Appendix D.2). Threads parallelise across queries, so multi-thread figures are throughput, not single-query latency. Multi-thread (1t / 8t) breakdown in Appendix Table 18.

Dataset	N	**Ov@K**	**Cov**	**Full (ms)**	**CB (ms)**	**Sp. vs Full**	**vs m-cpu**
ArguAna	8.7 K	0.95 / 0.96	14% / 20%	284 / 289	**23 / 36**	12.3× / 8.0×	4.6× / 2.8×
SciDocs	25.7 K	0.92 / 0.93	14% / 18%	821 / 832	**68 / 93**	12.1× / 8.9×	6.0× / 4.4×
MM	8.6 K	0.99 / 0.97	21% / 47%	1,636 / 1,657	**103 / 170**	15.9× / 9.7×	4.0× / 2.4×
HotpotQA	500 K	0.90 / 0.94	13% / 14%	7,394 / 7,215	**660 / 766**	11.2× / 9.4×	4.7× / 4.3×
Mean	—	—	—	—	—	**12.9× / 9.0×**	**4.8× / 3.5×**

axes stack with Col-Bandit's cell-skipping multiplicatively: *fewer dimensions* × *fewer tokens* × *fewer cells*.

Choosing the calibration knob α_{ef}. $\alpha_{\text{ef}} \in (0, 1]$ is the single continuous Pareto knob: smaller values tighten the decision radius and cut more aggressively. Setting α_{ef}=1 recovers the unshrunk empirical Bernstein–Serfling radius (the δ-PAC corner of Theorem 1). Table 20 gives operating-point ranges across the four BEIR text corpora at K=5: the aggressive α_{ef}=0.1 holds ∼13% coverage but drops Overlap@5 to 0.76 on the toughest corpus, the deployed α_{ef}=0.2 keeps 13–14% coverage at Overlap@5 ≥ 0.92, and the α_{ef}=1 setting reaches Overlap@5=1.00 at 28–59% coverage. Practitioners pick along this Pareto: 0.2 when speed dominates, 0.3 for fidelity headroom, 1.0 for the PAC guarantee. The headline-speedup chart at each $\alpha_{\text{ef}} \in \{0.1, 0.3, 1.0\}$ is in Appendix B.3 (Figures 4, 5, 6).

6 Conclusion

We presented Col-Bandit, an adaptive framework for accelerating late-interaction reranking at query time by selectively revealing MaxSim cells until the top-K set stabilizes. Across BEIR and REAL-MM-RAG, Col-Bandit consistently exposes substantial redundancy in dense late-interaction scoring, reducing MaxSim FLOPs by up to ∼ **8**× and translating these savings to a **7.0**×/**4.6**× (1/8 thread) wall-clock speedup on a server CPU and **12.9**×/**10.3**× (1/8 thread) on Apple M1 Max (four-corpus subset), while preserving high overlap with exhaustive reranking (Ov@5 ≥ 0.90 on every corpus tested, and ≥ 0.96 on CPU-S for every corpus but HotpotQA-500 K). A single calibration knob, α_{ef} (Eq. 7), provides a practical control over the quality–compute trade-off and yields strong Pareto frontiers against non-adaptive baselines. The α_{ef}=1 corner recovers a δ-PAC guarantee (Theorem 1, scoped in Remark 2). Col-Bandit is a drop-in reranking layer requiring no retraining or index changes, making it easy to deploy on top of standard search pipelines.

Limitations

Col-Bandit has three main limitations. **(i) Hardware scope.** Wall-clock evaluation is restricted to two CPU classes (an AVX2 server and a NEON laptop). We have not built a GPU-native fused kernel, so the GPU comparison uses an unfused PyTorch baseline. **(ii) Fidelity ceiling.** Col-Bandit estimates the exhaustive MaxSim top-K. It cannot exceed the quality of the underlying scorer, only recover it at lower cost. **(iii) Theory vs. deployment.** The δ-PAC guarantee (Theorem 1) holds at the δ-PAC corner $\alpha_{\mathrm{ef}}{=}1$ with the full Bernstein–Serfling radius. The deployed $\alpha_{\mathrm{ef}}{=}0.2$ uses a calibrated, simplified radius (pre-factor $c{=}1$, lower-order term omitted) and makes elimination decisions on int8-quantised estimates, so it carries no formal certificate, trading it for the cost–fidelity operating point we report. Separately, Quora's end-task metrics (nDCG/Recall/MRR) are unavailable in our harness (a qrels/doc-id mapping issue), so its task-metric averages are excluded throughout. Overlap@K, our primary fidelity metric, is unaffected.

Ethics Statement

This work accelerates an existing retrieval primitive (MaxSim) on public benchmarks (BEIR, REAL-MM-RAG). It introduces no new datasets, models, or human-subjects data. The only dual-use consideration is the generic one shared by any retrieval-efficiency method (faster search of an already indexed corpus). No new model capability is introduced.

References

Jean-Yves Audibert and Sébastien Bubeck. 2010. Best arm identification in multi-armed bandits. In *COLT-23th Conference on learning theory-2010*, pages 13–p.

Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256.

Rémi Bardenet and Odalric-Ambrym Maillard. 2015. Concentration inequalities for sampling without replacement. *Bernoulli*, 21(3):1361–1385.

Andrei Z Broder, David Carmel, Michael Herscovici, Aya Soffer, and Jason Zien. 2003. Efficient query evaluation using a two-level retrieval process. In *Proceedings of the twelfth international conference on Information and knowledge management*, pages 426–434.

Shouyuan Chen, Tian Lin, Irwin King, Michael R Lyu, and Wei Chen. 2014. Combinatorial pure exploration of multi-armed bandits. *Advances in neural information processing systems*, 27.

Benjamin Clavié, Antoine Chaffin, and Griffin Adams. 2024. Reducing the footprint of multi-vector retrieval with minimal performance impact via token pooling. *arXiv preprint arXiv:2409.14683*.

Benjamin Clavié and Sean Lee. 2025. maxsim-cpu: Maximising Maxsim Efficiency.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S Weld. 2020. Specter: Document-level representation learning using citation-informed transformers. *arXiv preprint arXiv:2004.07180*.

Laxman Dhulipala, Majid Hadian, Rajesh Jayaram, Jason Lee, and Vahab Mirrokni. 2024. MUVERA: Multi-vector retrieval via fixed dimensional encodings. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*.

Shuai Ding and Torsten Suel. 2011. Faster top-k document retrieval using block-max indexes. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 993–1002.

Joshua Engels, Benjamin Coleman, Vihan Lakshman, and Anshumali Shrivastava. 2023. DESSERT: An efficient algorithm for vector set search with vector set queries. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*.

Manuel Faysse, Hugues Sibille, Tony Wu, Bilel Omrani, Gautier Viaud, Céline Hudelot, and Pierre Colombo. 2024. Colpali: Efficient document retrieval with vision language models. *arXiv preprint arXiv:2407.01449*.

Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021. A white box analysis of colbert. In *European Conference on Information Retrieval*, pages 257–263. Springer.

Michael Günther, Saba Sturua, Mohammad Kalim Akram, Isabelle Mohr, Andrei Ungureanu, Bo Wang, Sedigheh Eslami, Scott Martens, Maximilian Werk, Nan Wang, and 1 others. 2025. jina-embeddings-v4: Universal embeddings for multimodal multilingual retrieval. In *Proceedings of the 5th Workshop on Multilingual Representation Learning (MRL 2025)*, pages 531–550.

Elias Jääsaari, Ville Hyvönen, and Teemu Roos. 2026. LEMUR: Learned multi-vector retrieval. *arXiv preprint arXiv:2601.21853*.

Rohan Jha, Bo Wang, Michael Günther, Georgios Mastrapas, Saba Sturua, Isabelle Mohr, Andreas Koukounas, Mohammad Kalim Akram, Nan Wang, and Han Xiao. 2024. Jina-colbert-v2: A general-purpose multilingual late interaction retriever. *arXiv preprint arXiv:2408.16672*.

Shivaram Kalyanakrishnan, Ambuj Tewari, Peter Auer, and Peter Stone. 2012. Pac subset selection in stochastic multi-armed bandits. In *Proceedings of the 29th International Conference on Machine Learning*, pages 655–662.

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.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, and 1 others. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Carlos Lassance, Maroua Maachou, Joohee Park, and Stéphane Clinchant. 2021. A study on token pruning for colbert. *arXiv preprint arXiv:2112.06540*.

Daniel LeJeune, Richard G. Baraniuk, and Reinhard Heckel. 2019. Adaptive estimation for approximate k-nearest-neighbor computations. *CoRR*, abs/1902.09465.

Sean MacAvaney, Antonio Mallia, and Nicola Tonellotto. 2025. Efficient constant-space multi-vector retrieval. In *European Conference on Information Retrieval*, pages 237–245. Springer.

Keshav Santhanam, Omar Khattab, Christopher Potts, and Matei Zaharia. 2022a. PLAID: An efficient engine for late interaction retrieval. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, CIKM '22, pages 1747–1756, New York, NY, USA. Association for Computing Machinery.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022b. 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, Seattle, United States. Association for Computational Linguistics.

Jan Luca Scheerer, Matei Zaharia, Christopher Potts, Gustavo Alonso, and Omar Khattab. 2025. Warp: An efficient engine for multi-vector retrieval. In *Proceedings of the 48th international ACM SIGIR conference on research and development in information retrieval*, pages 2504–2512.

Chengshuai Shi, Kun Yang, Jing Yang, and Cong Shen. 2024. Best arm identification for prompt learning under a limited budget. *arXiv preprint arXiv:2402.09723*.

IBM Research Team. 2025a. Granite-vision-3.3-2b-embedding.

Nomic Team. 2025b. Nomic embed multimodal: Interleaved text, image, and screenshots for visual document retrieval.

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*.

Nicola Tonellotto and Craig Macdonald. 2021. Query embedding pruning for dense retrieval. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 3453–3457.

Ash Vardanian. NumKong: 2000 Mixed Precision Kernels For All.

Henning Wachsmuth, Shahbaz Syed, and Benno Stein. 2018. Retrieval of the best counterargument without prior topic knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 241–251.

Xiao Wang, Craig Macdonald, Nicola Tonellotto, and Iadh Ounis. 2023. Reproducibility, replicability, and insights into dense multi-representation retrieval models: from colbert to col. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2552–2561.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, and 1 others. 2025. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2526–2547.

Navve Wasserman, Roi Pony, Oshri Naparstek, Adi Raz Goldfarb, Eli Schwartz, Udi Barzelay, and Leonid Karlinsky. 2025. Real-mm-rag: A real-world multi-modal retrieval benchmark. *arXiv preprint arXiv:2502.12342*.

Mengyao Xu, Gabriel Moreira, Ronay Ak, Radek Osmulski, Yauhen Babakhin, Zhiding Yu, Benedikt Schifferer, and Even Oldridge. 2025. Llama nemoretriever colembed: Top-performing text-image retrieval model. *arXiv:2507.05513*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2369–2380.

Jin Peng Zhou, Christian Walder, and 1 others. 2024. On speeding up language model evaluation. In *International Conference on Learning Representations (ICLR)*.

A Details of Variance-Adaptive Radius

Empirical Standard Deviation. The empirical standard deviation $\widehat{\sigma}_i$ used in the standard variance bound is calculated over the set of observed tokens $\mathcal{O}_i$:

$$\widehat{\sigma}_i^2 = \frac{1}{n_i - 1} \sum_{t \in \mathcal{O}_i} \left(H_{i,t} - \widehat{\mu}_i\right)^2. \tag{10}$$

In the edge case where $n_i \leq 1$, the variance is undefined; we strictly set $r_i^{\text{eff}} = +\infty$ and rely solely on the deterministic hard bounds.

Finite Population Correction (ρ_n). The term ρ_{n_i} in Eq. (7) accounts for sampling without replacement from a finite set of size T. It is defined piecewise as:

$$\rho_n \triangleq \begin{cases} 1 - \dfrac{n-1}{T}, & n \leq T/2, \\ \left(1 - \dfrac{n}{T}\right)\left(1 + \dfrac{1}{n}\right), & n > T/2. \end{cases} \tag{11}$$

This formulation ensures that the confidence interval shrinks faster than standard Bernstein bounds as $n \to T$. Specifically, when $n = T$, the term $(1 - n/T)$ becomes zero, collapsing the radius entirely as required for a fully observed document.

Bernstein–Serfling Constant c and $O(1/n)$ Bias Term. The constant c in Eq. (7) is the standard absolute Bernstein–Serfling pre-factor of Bardenet and Maillard (2015, Theorem 4.3). We treat it as c=1 in the implementation, since any difference (along with the $O(1/n)$ bias term standardly omitted from the simplified radius) is absorbed by the calibration factor α_{ef}. Moreover, the operative quantity is the *hybrid* interval $[\text{LCB}_i, \text{UCB}_i]$ (Eq. 8), which intersects the empirical radius with the deterministic support bounds $[LB_i^{\text{hard}}, UB_i^{\text{hard}}]$; the latter tend to be the binding constraint in the small-n regime where the omitted $O(1/n)$ term is largest, so in practice they cushion much of its effect. While the stopping time is adaptive, the procedure requires full separation of the top-K set, making it substantially less sensitive to optional-stopping risks than classical sequential hypothesis tests.

A.1 Reranking Setting

Our evaluation operates in the standard *full-corpus reranking* regime: given a query $Q = \{\mathbf{q}_1, \ldots, \mathbf{q}_T\}$ and a candidate set $\mathcal{D} = \{d_1, \ldots, d_N\}$, the goal is to return the top-K documents according to exact ColBERT-style late-interaction scoring (Eq. 5). The candidate set is the entire corpus throughout the main results (e.g. N=2.68M for NQ); first-stage retrievers (PLAID inverted index, MUVERA-FDE) are an orthogonal axis treated as black-box upstream components, so $\mathcal{D}$ may be replaced by their output without changing the algorithm. Col-Bandit acts on the $N \times T$ MaxSim matrix and returns the same top-K that exhaustive Full-MaxSim would produce on the same $\mathcal{D}$, with δ-PAC certification at α_{ef}=1 (Theorem 1).

A.2 Datasets and Models

We evaluate Col-Bandit on five widely used text retrieval datasets from the BEIR benchmark (Thakur et al., 2021): **ArguAna** (Wachsmuth et al., 2018), **Quora** (Thakur et al., 2021), **SciDocs** (Cohan et al., 2020), **NQ** (Kwiatkowski et al., 2019), and **HotPotQA** (Yang et al., 2018). We use two state-of-the-art late-interaction text embedding models: **ColBERTv2**[3] (Santhanam et al., 2022b) and **Jina-ColBERT-v2**[4] (Jha et al., 2024). Both models produce token embeddings of dimension $d = 128$ and use a fixed query token length of $T = 32$. In addition, we evaluate Col-Bandit on a visual document retrieval task using the **REAL-MM-RAG** (Wasserman et al., 2025) benchmark which include 4 subsets: FinReports, FinSlides, TechReports and TechSlides. In this setting, we employ the **Granite Vision Embedding 3.2**[5] (Team, 2025a) model, a vision-language embedding model that produces $d = 128$-dimensional token

[3] https://huggingface.co/lightonai/colbertv2.0
[4] https://huggingface.co/jinaai/jina-colbert-v2
[5] https://huggingface.co/ibm-granite/granite-vision-3.3-2b-embedding

Table 4: Evaluation datasets statistics. T_q: query token count; L_d: document token count (mean for text, fixed for vision). Quora end-task metrics (nDCG/Recall/MRR) are zero across *all* methods due to a known qrels/doc-id mapping issue in our harness; Overlap@K (the primary fidelity metric) is unaffected.

Dataset	**Corpus** N	**Queries**	T_q	L_d	**Encoder**	**Modality**
BEIR (language)						
ArguAna	8,674	1.4K	32	192	ColBERTv2, Jina-ColBERT-v2	Text
SciDocs	25,657	1K	32	187	ColBERTv2, Jina-ColBERT-v2	Text
HotpotQA-500K	500,000	1K	32	68	ColBERTv2, Jina-ColBERT-v2	Text
Quora	522,931	5K	32	32	ColBERTv2, Jina-ColBERT-v2	Text
NQ-2.68M	2,681,468	3.5K	32	68	ColBERTv2	Text
REAL-MM-RAG (multimodal)						
FinSlides	2,280	1K	32	729	Granite Vision Embedding 3.2	Image+Text
FinReport	2,687	853	32	729	Granite Vision Embedding 3.2	Image+Text
TechSlides	1,963	1.4K	32	729	Granite Vision Embedding 3.2	Image+Text
TechReport	1,674	1.3K	32	729	Granite Vision Embedding 3.2	Image+Text

embeddings, with variable-length query representations and 729 document tokens per image. Table 4 summarizes the key statistics of all evaluation datasets.

Licensing and intended use. All evaluation artifacts are publicly released research resources, used here in accordance with their intended use. The BEIR corpora (Thakur et al., 2021) and the REAL-MM-RAG benchmark (Wasserman et al., 2025) are distributed for research under their respective licenses, and the pretrained encoders (ColBERTv2 (Santhanam et al., 2022b), Jina-ColBERT-v2 (Jha et al., 2024), and Granite Vision Embedding 3.2 (Team, 2025a)) are released via Hugging Face under their respective model licenses. Col-Bandit is a purely inference-time reranking layer over these artifacts: it introduces no new datasets or model weights and redistributes none of the underlying data. The one artifact we contribute, the CB-NK kernel, will be released as open-source software for research and general use (Appendix D.2).

A.3 Compared Methods

All compared methods operate, like Col-Bandit, on the Stage-2 candidate set (Appendix A.1) and target the same exhaustive late-interaction top-K on that set. We compare against two zero-shot query-time pruning baselines that consume the same (N, L_d, d) corpus tensor as Col-Bandit:

Doc-Uniform (γ-sweep). Doc-Uniform is a non-adaptive "blind random scoring budget": for each query, sample $\lceil \gamma T \rceil$ cells uniformly at random per document ($\approx \gamma \cdot N \cdot T$ cells in total, matching Algorithm 2), score them, and predict the top-K by the sum of the revealed cells per document. We sweep $\gamma \in \{0.05, 0.10, 0.15, \ldots, 0.50, 0.70, 0.80, 0.90, 1.00\}$ and report the resulting Overlap@K vs. coverage curve. This baseline is the *redundancy upper bound*: if Col-Bandit cannot beat Doc-Uniform on the quality–coverage Pareto, then nothing query-adaptive is happening and we are merely paying for some random subset of cells. This is exactly the unstructured-budget reference we use in the §4/§5 Pareto plots.

Ball-carving (τ-sweep). Ball-carving is the procedure described in MUVERA, Appendix C.3 (Dhulipala et al., 2024), and is the strongest published query-time pruning baseline that operates on the same (N, L_d, d) corpus tensor we score. In contrast to Doc-Uniform, Ball-carving is a *query-side* compression: it greedily clusters the T query tokens into k groups using a cosine-similarity threshold τ, replaces each cluster $\mathcal{C}_i$ with a centroid $c_i = \sum_{q \in \mathcal{C}_i} q$, and rescores every document via Chamfer/MaxSim over the k centroids: $\mathrm{Score}(d) = \sum_{i=1}^{k} \max_{j \le L_d} \langle c_i, e_j^{(d)} \rangle$. Coverage is reported as k/T: smaller τ collapses more tokens into fewer, larger clusters (lower coverage), while $\tau = 1$ keeps every token as its own cluster and recovers exhaustive Chamfer. We sweep $\tau \in \{0.40, 0.50, 0.60, 0.70, 0.80, 0.90\}$ and average k/T across queries to place the operating point on the coverage axis. Our implementation matches MUVERA §3 / App. C.3 exactly.

Algorithm 2 Doc-Uniform (Static Random Reveal)

Require: Docs $\mathcal{D}$, Query Q, K, $\gamma \in [0, 1]$
$N \leftarrow |\mathcal{D}|$, $B \leftarrow \lceil \gamma T \rceil$ ▷ Cells per row
$\Omega \leftarrow \emptyset$, $H \in \mathbb{R}^{N \times T}$
for $i = 1$ **to** N **do**
  Sample $\mathcal{R}_i \subseteq [T]$ uniformly ▷ w/o replacement
  s.t. $|\mathcal{R}_i| = B$
  for each $t \in \mathcal{R}_i$ **do**
    $H_{i,t} \leftarrow h(d_i, t)$ ▷ Reveal MaxSim
    $\Omega \leftarrow \Omega \cup \{(i, t)\}$
  end for
  $\widetilde{S}_i \leftarrow \sum_{t \in \mathcal{R}_i} H_{i,t}$ ▷ Static score
end for
return $\arg\operatorname{topK}_{i \in [N]} \widetilde{S}_i$

Algorithm 3 Ball-carving query compression (MUVERA Appendix C.3)

Require: Query $Q = (q_1, \ldots, q_T) \in \mathbb{R}^{T \times d}$, threshold $\tau \in (0, 1]$
Require: Docs $\mathcal{D}$ with embeddings $(e_1^{(d)}, \ldots, e_{L_d}^{(d)})$, K
// Step 1: greedy query-side clustering
$\hat{q}_i \leftarrow q_i / \|q_i\|$ **for all** $i \in [T]$
$\mathcal{R} \leftarrow \{1, \ldots, T\}$, $k \leftarrow 0$
while $\mathcal{R} \neq \emptyset$ **do**
  $k \leftarrow k + 1$, $p \leftarrow \min \mathcal{R}$ ▷ pivot
  $\mathcal{C}_k \leftarrow \{i \in \mathcal{R} : \langle \hat{q}_p, \hat{q}_i \rangle \geq \tau\}$
  $c_k \leftarrow \sum_{i \in \mathcal{C}_k} q_i$
  $\mathcal{R} \leftarrow \mathcal{R} \setminus \mathcal{C}_k$
end while
// Step 2: rescore with one MaxSim term per cluster
for each $d \in \mathcal{D}$ **do**
  $\operatorname{Score}(d) \leftarrow \sum_{i=1}^{k} \max_{j \leq L_d} \langle c_i, e_j^{(d)} \rangle$
end for
return $\arg\operatorname{topK}_{d \in \mathcal{D}} \operatorname{Score}(d)$

B Extended Experimental Results

B.1 Universal Efficiency at the 90% Threshold

The body table (Table 1) reports the stricter 95% near-lossless threshold. For comparability with prior work that reports the looser 90% retention threshold, we provide the companion table below; the layout is identical to the body table apart from the threshold.

B.2 Detailed Efficiency Results per Dataset

In the main text (Table 1), we present efficiency metrics averaged across the BEIR and REAL-MM-RAG suites. The tables in this subsection break those averages down per dataset, organized by metric (Overlap, nDCG, Recall, MRR) and target rank ($K \in \{5, 10, 50\}$). Each cell reports the coverage budget (%) at which Col-Bandit reaches **90%** (white) and **95%** (gray) of Full-MaxSim's metric on that dataset. *Note:* cells marked "n/a" indicate that the corresponding (encoder, dataset) sweep was not run; in particular, the Jina-ColBERTv2 sweep covers four of the five BEIR text corpora (NQ-2.68M not benchmarked).

This detailed view confirms that the efficiency gains of Col-Bandit are robust across diverse data distributions: Col-Bandit consistently outperforms the baselines on every individual dataset, with the exact magnitude varying with document length and query difficulty.

The eight tables below cover the four metrics (Overlap@K, nDCG@K, Recall@K, MRR@K) on the BEIR text suite (top half) and the REAL-MM-RAG multimodal suite (bottom half).

B.3 Headline Speedup at Other α_{ef} and K

This subsection extends the headline figure from the main paper (Figure 1, which fixes the deployed knob α_{ef}=0.2, K=5) by sweeping the calibration knob α_{ef} and the target rank K while holding everything else identical (CPU-S = AMD EPYC 7763, 1/8/16 threads, BEIR + REAL-MM-RAG, ColBERTv2 / GVE). Each figure uses the same bar/diamond layout as the body figure: bars are CB-vs-Full speedup at 1/8/16 threads per corpus; right-axis purple diamonds are Overlap@K vs. Full-MaxSim's exhaustive top-K.

α_{ef} sweep at K=5. Smaller α_{ef} tightens the decision radius and eliminates more aggressively, trading lower Overlap@5 for higher speedup. The deployed default (α_{ef}=0.2, body figure) is bracketed by Figure 4 (aggressive, α_{ef}=0.1), Figure 5 (conservative, α_{ef}=0.3), and Figure 6 (δ-PAC corner α_{ef}=1, Theorem 1). Across the three settings the qualitative ranking of corpora is preserved; the deployed setting strikes the practical Pareto trade-off between speedup and Overlap@5.

K sweep at α_{ef}=0.2. Holding the deployed knob fixed and growing K to a recall-class setting (K=100). Figure 7 confirms the sub-linear-in-K behaviour of Col-Bandit reported quantitatively in §5.3 (Table 19):

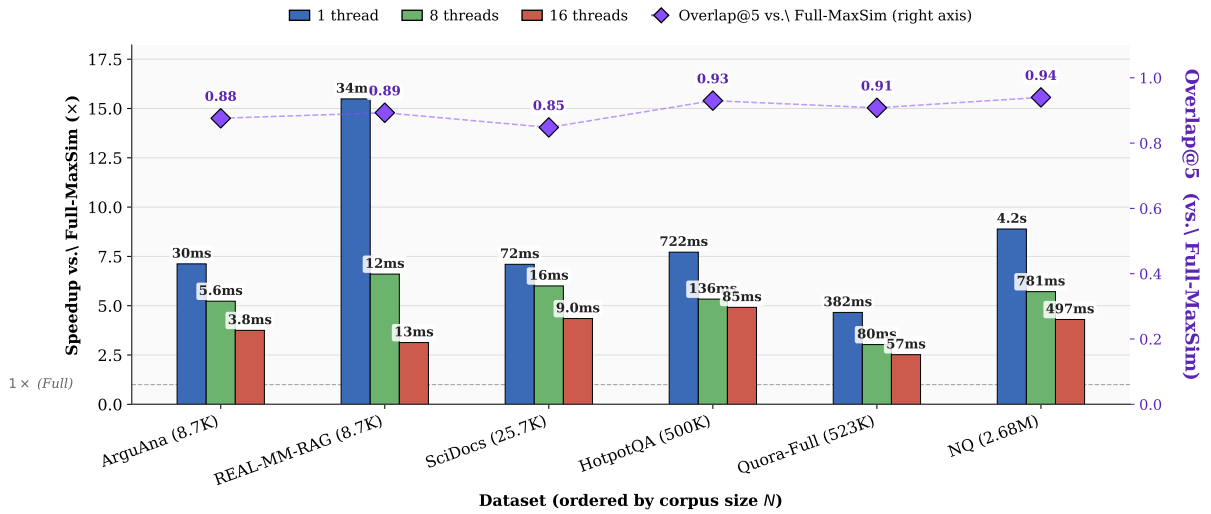

Figure 4: Wall-clock speedup at α_{ef}=0.1 (aggressive corner): higher speedup than the deployed knob but Ov@5 falls as low as 0.76 on the toughest corpus (full per-corpus ranges in Table 20).

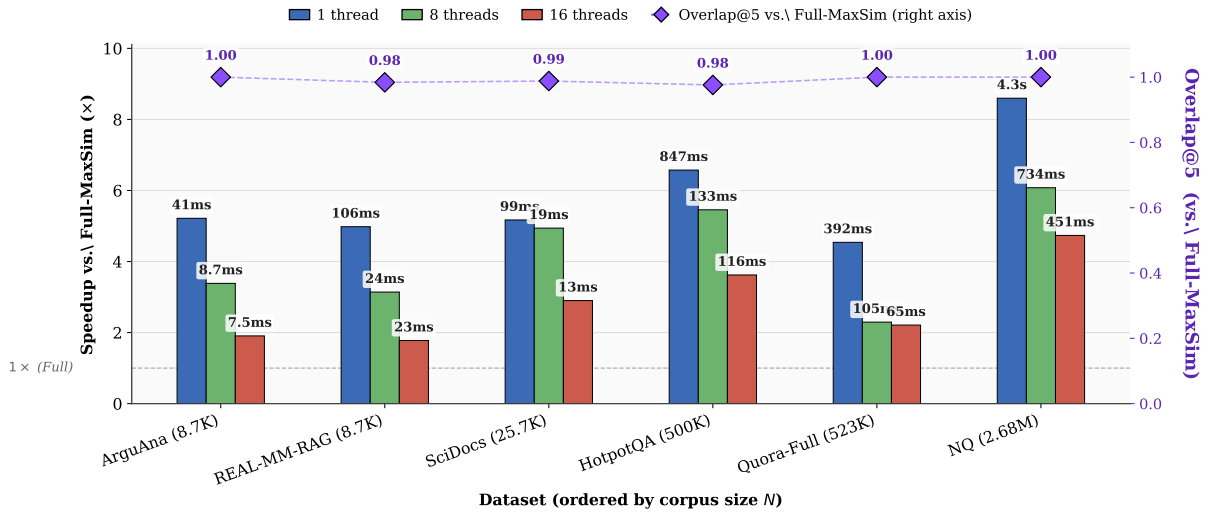

Figure 5: Wall-clock speedup at α_{ef}=0.3 (conservative): a slightly tighter knob than deployed; Ov@5 $\geq$ 0.98 on every corpus at modestly reduced speedup.

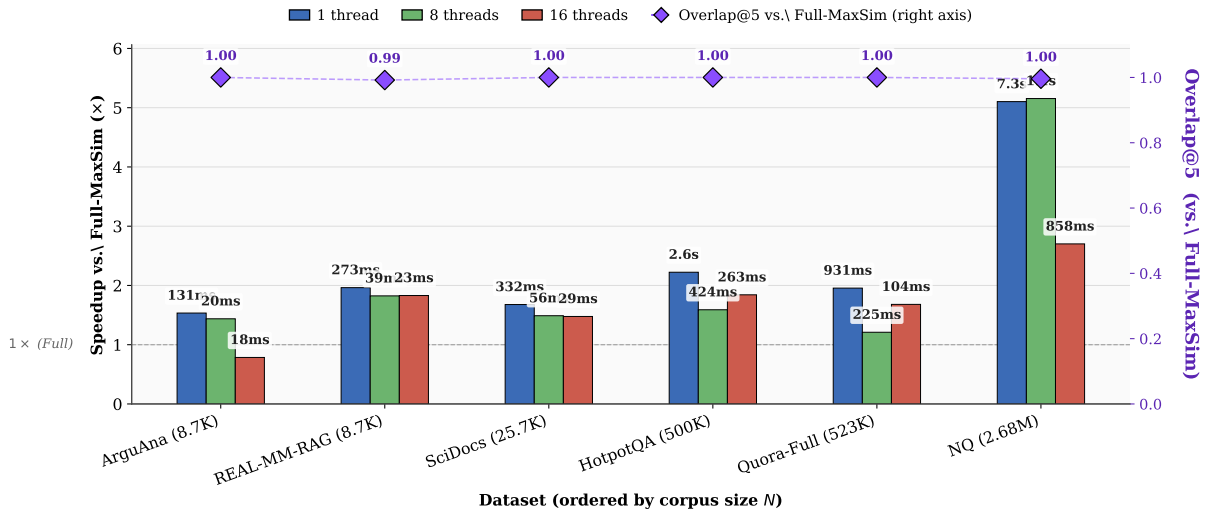

Figure 6: Wall-clock speedup at α_{ef}=1 (δ-PAC corner of Theorem 1): Ov@5=1.0 on every text corpus, at the cost of $\sim$28–59% coverage. Even at the PAC-valid setting Col-Bandit delivers nontrivial speedups over Full-MaxSim.

Table 5: **Universal Efficiency Analysis at the 90% threshold (Appendix companion to Table 1).** Each cell reports **mean coverage % (std) / savings** × required to recover **90%** of Full-MaxSim's own Overlap@K / nDCG@K per dataset; savings = 100%/mean coverage. The body table reports the stricter 95% near-lossless threshold; this appendix table provides the looser 90% threshold for comparability with prior work.

Method	Overlap@5	Overlap@50	nDCG@5	nDCG@50
ColBERTv2 (BEIR)				
Doc-Uniform	94% (7.5) / 1.1×	92% (7.0) / 1.1×	38% (11.7) / 2.7×	25% (4.4) / 4.0×
Ball-carving	34% (11.1) / 2.9×	32% (8.9) / 3.2×	21% (3.7) / 4.8×	19% (5.9) / 5.2×
Col-Bandit (Ours)	**13% (0.9) / 7.4×**	**15% (2.5) / 6.6×**	**13% (0.1) / 7.9×**	**13% (0.5) / 7.6×**
Jina-ColBERTv2 (BEIR)				
Doc-Uniform	93% (7.8) / 1.1×	93% (7.8) / 1.1×	52% (26.6) / 1.9×	45% (24.4) / 2.2×
Ball-carving	37% (9.7) / 2.7×	34% (6.2) / 2.9×	16% (3.8) / 6.3×	16% (3.8) / 6.3×
Col-Bandit (Ours)	**17% (11.2) / 5.9×**	**28% (14.6) / 3.6×**	**7% (1.2) / 13.5×**	**9% (2.1) / 10.9×**
Granite-Vision-Embedding (REAL-MM-RAG)				
Doc-Uniform	83% (4.3) / 1.2×	83% (4.3) / 1.2×	34% (11.7) / 2.9×	27% (8.8) / 3.7×
Ball-carving	46% (2.1) / 2.2×	39% (5.3) / 2.6×	19% (5.3) / 5.2×	19% (5.3) / 5.2×
Col-Bandit (Ours)	**16% (4.8) / 6.1×**	**26% (5.4) / 3.9×**	**10% (1.5) / 10.4×**	**12% (1.3) / 8.6×**

the speedup degrades modestly relative to the body K=5 headline because the elimination phase has to discriminate among more candidates, but the Overlap@100 stays ≥ 0.93 on every corpus.

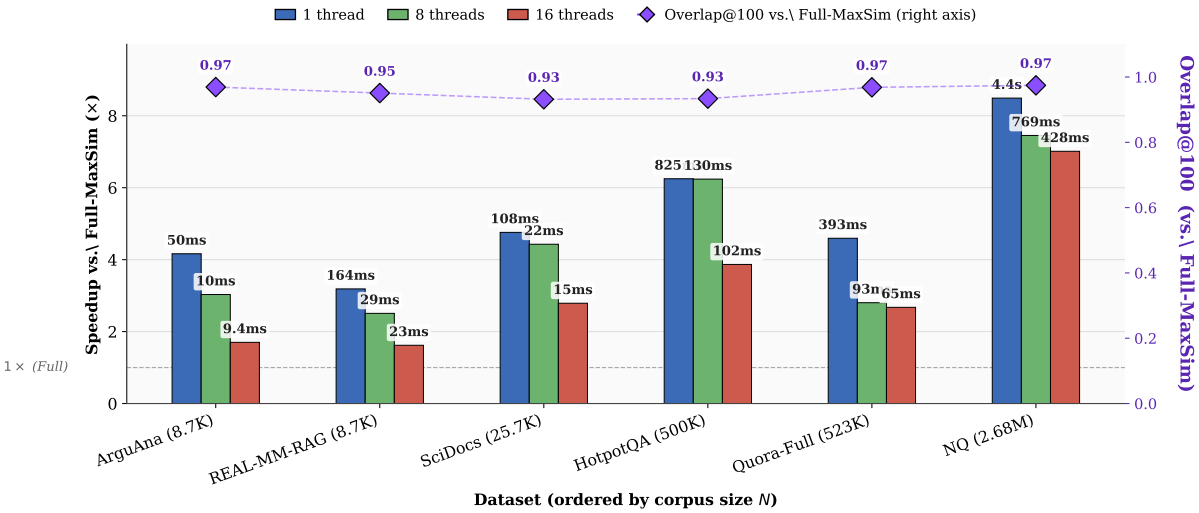

Figure 7: Wall-clock speedup at the deployed α_{ef}=0.2, K=100 (recall-class setting): Col-Bandit retains substantial speedup at the larger top-K target with Ov@100 ≥ 0.93 on every corpus.

B.4 Extended Retrieval Effectiveness (Top-1 Analysis)

The Top-1 ranking regime (Recall@1, nDCG@1, MRR@1) is a stricter test of fidelity: a single misidentified document costs the full point. Table 14 reports retrieval effectiveness at varying coverage levels averaged across BEIR and REAL-MM-RAG. The Top-1 trends mirror those at $K = 5$ (Table 1): Col-Bandit maintains near-lossless performance compared to the Full ColBERT baseline, even when pruning significantly more aggressively than non-adaptive methods. At lower coverage budgets, the gap between Col-Bandit and the non-adaptive baselines (Doc-Uniform and Ball-carving) widens, highlighting the necessity of variance-aware sampling for correctly identifying the single best document with high confidence.

B.5 Per-Corpus Retrieval Effectiveness on REAL-MM-RAG

Table 15 breaks retrieval effectiveness down per corpus on the four REAL-MM-RAG sub-corpora (FinReport, FinSlides, TechReport, TechSlides), reporting Recall@5, nDCG@5, and MRR@5 at two coverage operating points ($\sim 20\%$ and $\sim 40\%$). At the $\sim 20\%$ point Col-Bandit retains $\geq 97.8\%$ of Full ColBERT's Recall@5/nDCG@5/MRR@5 on every corpus, versus 78–95% for Ball-carving and

Table 6: **Per-dataset Efficiency Analysis: Overlap@K (BEIR).** We report the coverage budget required to achieve **90%** (white) and **95%** (gray) of Full-MaxSim's **Overlap@K** per corpus. Under **Average**, we report **mean coverage (std)** across datasets, and **Savings (vs. Full)** is the compute reduction factor relative to full reranking (i.e., $100\%/$**Mean**).

Task Domain	**Text Retrieval Benchmarks (BEIR)**										**Average**			
Method	**SciDocs**		**Quora**		**NQ**		**HotpotQA**		**ArguAna**		*Mean (std)*		*Savings (vs. Full)*	
$K = 5$														
ColBERTv2														
Doc-Uniform	100%	100%	81%	100%	81%	91%	100%	100%	91%	100%	91% (8.4)	98% (3.8)	1.10×	1.02×
Ball-carving	31%	36%	22%	27%	21%	28%	47%	61%	44%	60%	33% (10.7)	42% (14.9)	3.02×	2.36×
Col-Bandit (Ours)	**14%**	**17%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**15%**	**15%**	**13% (0.9)**	**14% (1.7)**	**7.42×**	**7.06×**
Jina-ColBERT-V2														
Doc-Uniform	100%	100%	91%	100%	*n/a*		100%	100%	81%	100%	93% (7.8)	100% (0.0)	1.08×	1.00×
Ball-carving	33%	100%	24%	24%	*n/a*		40%	100%	51%	51%	37% (9.7)	69% (32.6)	2.69×	1.45×
Col-Bandit (Ours)	**19%**	**35%**	**7%**	**11%**	*n/a*		**34%**	**49%**	**7%**	**8%**	**17% (11.2)**	**26% (16.9)**	**5.95×**	**3.92×**
$K = 10$														
ColBERTv2														
Doc-Uniform	100%	100%	91%	100%	81%	100%	100%	100%	91%	100%	92% (7.0)	100% (0.0)	1.08×	1.00×
Ball-carving	29%	36%	22%	27%	21%	28%	38%	61%	50%	60%	32% (10.9)	42% (14.9)	3.10×	2.36×
Col-Bandit (Ours)	**15%**	**15%**	**13%**	**13%**	**13%**	**13%**	**13%**	**15%**	**15%**	**19%**	**14% (1.2)**	**15% (2.4)**	**7.28×**	**6.68×**
Jina-ColBERT-V2														
Doc-Uniform	100%	100%	91%	100%	*n/a*		100%	100%	70%	100%	90% (12.1)	100% (0.0)	1.11×	1.00×
Ball-carving	33%	100%	24%	100%	*n/a*		40%	100%	39%	51%	34% (6.2)	88% (21.3)	2.94×	1.14×
Col-Bandit (Ours)	**60%**	**60%**	**16%**	**16%**	*n/a*		**34%**	**69%**	**11%**	**11%**	**30% (19.0)**	**39% (25.7)**	**3.31×**	**2.57×**
$K = 50$														
ColBERTv2														
Doc-Uniform	100%	100%	91%	100%	81%	100%	100%	100%	91%	100%	92% (7.0)	100% (0.0)	1.08×	1.00×
Ball-carving	29%	36%	22%	27%	21%	23%	38%	61%	44%	60%	31% (8.9)	41% (15.9)	3.23×	2.42×
Col-Bandit (Ours)	**17%**	**22%**	**14%**	**15%**	**13%**	**13%**	**14%**	**16%**	**19%**	**26%**	**15% (2.4)**	**18% (4.8)**	**6.54×**	**5.46×**
Jina-ColBERT-V2														
Doc-Uniform	100%	100%	91%	100%	*n/a*		100%	100%	81%	100%	93% (7.8)	100% (0.0)	1.08×	1.00×
Ball-carving	33%	100%	24%	100%	*n/a*		40%	100%	39%	51%	34% (6.2)	88% (21.3)	2.94×	1.14×
Col-Bandit (Ours)	**39%**	**52%**	**15%**	**29%**	*n/a*		**45%**	**61%**	**12%**	**16%**	**28% (14.6)**	**39% (17.8)**	**3.58×**	**2.55×**

Doc-Uniform; the gap widens as the coverage budget tightens, underscoring the value of variance-aware sampling for high-fidelity ranking.

B.6 Runtime Overhead Breakdown

Table 16 decomposes per-query Col-Bandit latency into three stages: the MaxSim micro-kernel calls during elimination rounds (kernel internals in Appendix D.2); the elimination decision points themselves (LCB/UCB updates and survivor-set bookkeeping); and the final $K+M$ exhaustive-aligned rescore. Across the eight corpora profiled here (HotpotQA omitted from this profiling run) at the deployed knob, the kernel dominates ($\geq 96\%$ of total runtime); elimination plus rescore together stay under 4%. This is consistent with the speedup model used throughout the main text: the wallclock benefit comes from calling the kernel on $\mathrm{Cov}\%$ of the (d, t) cells rather than 100%, not from cheaper bookkeeping.

Deviations from the $100/\mathrm{Cov}\%$ heuristic. The realised speedup deviates from the first-order heuristic in both directions: thread scaling, memory traffic, and the elimination/rescore phases combine to make small corpora track the heuristic only loosely. For example, Quora at 13% Cov yields only $4.7\times$ speedup at 1 t (rescore overhead dominates relative to a small kernel call), while MM at 21% Cov yields $8.1\times$ at 1 t (the cell-skip pattern aligns favourably with the SIMD register tile). Tightening α_{ef} from the deployed 0.2 up to the $\alpha_{\mathrm{ef}}{=}1$ corner trades additional coverage for higher fidelity, as shown by the full Pareto frontier in Figure 8.

B.7 Multi-Thread Wall-Clock Scaling

The main-paper wall-clock tables (Tables 2 and 3) report single-thread latency for compactness. Tables 17 and 18 give the full 1t / 8t breakdown on CPU-S (AMD EPYC 7763, AVX2) and CPU-M1 (Apple M1 Max, NEON) respectively. Thread scaling is sub-linear because the $K+M$ rescore and elimination

Table 7: **Per-dataset Efficiency Analysis: Overlap@K (REAL-MM-RAG).** Coverage budget required to achieve **90%** (white) and **95%** (gray) of Full-MaxSim's **Overlap@K** per corpus, on the Granite-Vision-Embedding (GVE) multimodal benchmarks. **Mean (std)** averages across the four corpora; **Savings** is $100\%/$**Mean**.

Task Domain	**Multimodal (REAL-MM-RAG, GVE)**								**Average**			
Method	**FinSlides**		**FinReport**		**TechSlides**		**TechReport**		*Mean (std)*		*Savings (vs. Full)*	
$K = 5$												
Doc-Uniform	81%	100%	91%	100%	81%	100%	81%	100%	83% (4.3)	100% (0.0)	1.20×	1.00×
Ball-carving	44%	75%	44%	74%	47%	75%	49%	79%	46% (2.1)	76% (1.7)	2.18×	1.32×
Col-Bandit (Ours)	**19%**	**19%**	**12%**	**23%**	**12%**	**22%**	**23%**	**23%**	**16% (4.8)**	**22% (1.9)**	**6.11×**	**4.59×**
$K = 10$												
Doc-Uniform	91%	100%	91%	100%	91%	100%	81%	100%	88% (4.3)	100% (0.0)	1.13×	1.00×
Ball-carving	44%	75%	44%	74%	33%	47%	49%	79%	43% (5.9)	69% (12.9)	2.35×	1.46×
Col-Bandit (Ours)	**21%**	**21%**	**13%**	**27%**	**14%**	**25%**	**26%**	**26%**	**19% (5.5)**	**25% (2.3)**	**5.40×**	**4.01×**
$K = 50$												
Doc-Uniform	81%	100%	91%	100%	81%	100%	81%	91%	83% (4.3)	98% (4.1)	1.20×	1.02×
Ball-carving	44%	75%	44%	74%	33%	75%	34%	79%	39% (5.3)	76% (1.7)	2.58×	1.32×
Col-Bandit (Ours)	**35%**	**35%**	**21%**	**43%**	**23%**	**42%**	**24%**	**43%**	**26% (5.4)**	**41% (3.1)**	**3.85×**	**2.46×**

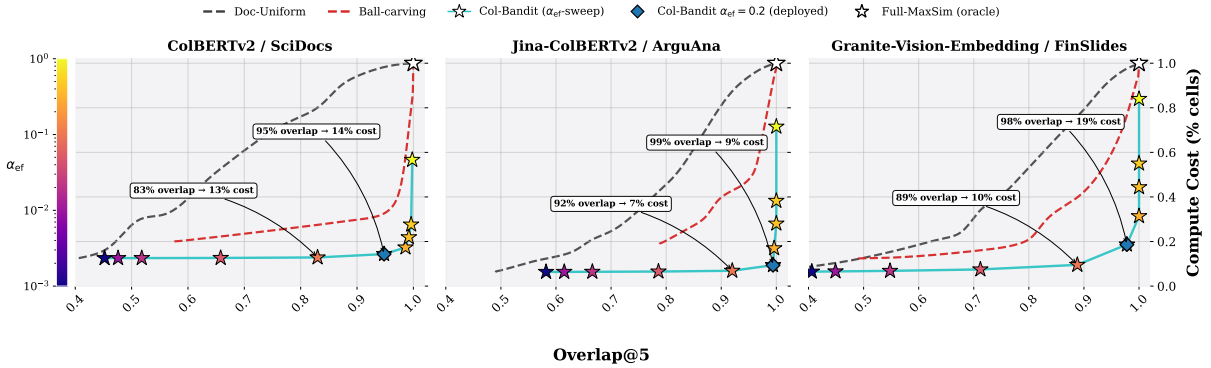

Figure 8: **Quality–coverage Pareto frontier (K=5).** Col-Bandit (sweeping α_{ef}) vs. Doc-Uniform and Ball-carving (MUVERA App. C.3) on ColBERTv2/SciDocs, Jina-ColBERTv2/ArguAna, and GVE/FinSlides. Col-Bandit dominates both baselines, reaching high fidelity at $\leq 20\%$ of the cell budget; deployed knob α_{ef}=0.2 marked, Full-MaxSim oracle at top-right.

bookkeeping do not parallelise as cleanly as the dense kernel sweep, but Col-Bandit retains a substantial multi-thread speedup on every corpus.

B.8 Operating-Point, K-Sensitivity, and Commodity-CPU Tables

These three tables support claims made in §5.2 and are placed here for space. Table 19 reports the K-sensitivity sweep ($K \in \{5, 100, 500, 1000\}$ on CPU-S, single thread) discussed under "Sub-linear scaling in K". Table 20 gives the operating-point guidance for the calibration knob α_{ef} referenced in "Choosing the calibration knob α_{ef}". Table 21 is the commodity-CPU-vs-GPU architecture-level pairing (16-thread Col-Bandit on CPU-S vs. A100 PyTorch Full-MaxSim) discussed under "Commodity CPU vs. GPU".

B.9 Full Three-Panel Pareto Frontier

The main paper (Figure 3) shows the GVE/FinSlides panel for space. Figure 8 gives the full three-panel comparison across one small-text (ColBERTv2/SciDocs), one large-text (Jina-ColBERTv2/ArguAna), and one multimodal (GVE/FinSlides) corpus.

B.10 Matryoshka Dimension Reduction

Jina-ColBERTv2's d=128 $\rightarrow$ d=64 Matryoshka projection halves the per-token storage with no impact on Col-Bandit's elimination dynamics. Table 22 sweeps the four BEIR text corpora at the deployed knob (α_{ef}=0.2, K=5). Coverage is essentially invariant to dimension (shifts ≤ 1.7 percentage points on three

Table 8: **Per-dataset Efficiency Analysis: nDCG@K (BEIR).** We report the coverage budget required to achieve **90%** (white) and **95%** (gray) of Full-MaxSim's **NDCG@K** per corpus. Under **Average**, we report **mean coverage (std)** across datasets, and **Savings (vs. Full)** is the compute reduction factor relative to full reranking (i.e., $100\%/$**Mean**). Quora is shown as *n/a* and excluded from the Average: its end-task metrics are zero across all methods under our harness (a qrels/doc-id mapping issue; Table 4), so per-corpus task-metric budgets are undefined there. Quora retrieval fidelity is unaffected and reported via Overlap@K (Table 6).

Task Domain	**Text Retrieval Benchmarks (BEIR)**									**Average**			
Method	**SciDocs**		**Quora**	**NQ**		**HotpotQA**		**ArguAna**		*Mean (std)*		*Savings (vs. Full)*	
$K = 5$													
ColBERTv2													
Doc-Uniform	41%	72%	*n/a*	22%	50%	50%	81%	50%	50%	41% (11.4)	63% (13.6)	2.45×	1.58×
Ball-carving	20%	26%	*n/a*	17%	17%	30%	34%	26%	44%	23% (5.1)	30% (10.0)	4.30×	3.31×
Col-Bandit (Ours)	**13%**	**14%**	*n/a*	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13% (0.0)**	**13% (0.4)**	**7.69×**	**7.55×**
Jina-ColBERT-V2													
Doc-Uniform	91%	100%	*n/a*	*n/a*		50%	70%	50%	50%	64% (19.3)	73% (20.5)	1.57×	1.36×
Ball-carving	15%	18%	*n/a*	*n/a*		20%	26%	19%	24%	18% (2.2)	23% (3.4)	5.56×	4.41×
Col-Bandit (Ours)	**10%**	**15%**	*n/a*	*n/a*		**7%**	**7%**	**7%**	**7%**	**8% (1.4)**	**10% (3.8)**	**12.50×**	**10.34×**
$K = 10$													
ColBERTv2													
Doc-Uniform	22%	41%	*n/a*	22%	50%	50%	81%	41%	72%	34% (12.2)	61% (16.1)	2.96×	1.64×
Ball-carving	20%	26%	*n/a*	12%	17%	23%	30%	26%	36%	20% (5.2)	27% (6.9)	4.94×	3.67×
Col-Bandit (Ours)	**13%**	**15%**	*n/a*	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13% (0.0)**	**14% (0.9)**	**7.69×**	**7.41×**
Jina-ColBERT-V2													
Doc-Uniform	70%	91%	*n/a*	*n/a*		50%	70%	41%	70%	54% (12.1)	77% (9.9)	1.86×	1.30×
Ball-carving	15%	18%	*n/a*	*n/a*		20%	26%	19%	30%	18% (2.2)	25% (5.0)	5.56×	4.05×
Col-Bandit (Ours)	**22%**	**60%**	*n/a*	*n/a*		**7%**	**34%**	**7%**	**7%**	**12% (7.1)**	**34% (21.6)**	**8.33×**	**2.97×**
$K = 50$													
ColBERTv2													
Doc-Uniform	22%	50%	*n/a*	22%	22%	50%	81%	31%	41%	31% (11.4)	48% (21.3)	3.20×	2.06×
Ball-carving	20%	26%	*n/a*	12%	17%	23%	30%	26%	26%	20% (5.2)	25% (4.8)	4.94×	4.04×
Col-Bandit (Ours)	**14%**	**14%**	*n/a*	**13%**	**13%**	**13%**	**14%**	**13%**	**14%**	**13% (0.4)**	**14% (0.4)**	**7.55×**	**7.27×**
Jina-ColBERT-V2													
Doc-Uniform	81%	91%	*n/a*	*n/a*		50%	50%	31%	70%	54% (20.6)	70% (16.7)	1.85×	1.42×
Ball-carving	15%	23%	*n/a*	*n/a*		20%	20%	19%	24%	18% (2.2)	22% (1.7)	5.56×	4.48×
Col-Bandit (Ours)	**13%**	**27%**	*n/a*	*n/a*		**10%**	**12%**	**8%**	**9%**	**10% (2.1)**	**16% (7.9)**	**9.68×**	**6.25×**

of four corpora; on Quora-Full coverage actually *drops* 5.9 pp at d=64 with no quality loss). Ov@5 retention is ≥ 0.95 on every cell; nDCG@5 retention is 100% on six of eight cells. The dimension and cell-skipping axes are orthogonal: halving d does not change which documents Col-Bandit eliminates.

B.11 Ward Token-Axis Pooling

Ward hierarchical clustering (Clavié et al., 2024) pools each document's L_d token embeddings into L_d/k cluster centroids ($k \in \{2, 4, 8\}$), with the clustering applied per-document on the actual non-padded token count. This is orthogonal to Col-Bandit: pooling reduces the number of cells per row at encode time; Col-Bandit then eliminates documents based on the (smaller) per-row score at query time. We benchmark the stack on CPU-S for SciDocs (text) and TechSlides (multimodal) at the deployed knob.

Table 23 reports both *Fidelity Ov*@5 (against the exhaustive top-5 on the pooled corpus) and *End-to-End Ov*@5 (against the exhaustive top-5 on the original unpooled corpus), with one row per pool factor showing Ward-Full alone (100% coverage) and one row showing Ward-Full + Col-Bandit. The two-row layout makes the loss decomposition explicit: Ward-Full's E2E gap captures *pooling-induced* information loss; the Δ from Ward-Full to Ward-Full + Col-Bandit captures Col-Bandit's *elimination* cost. The numbers show pooling dominates: across the six cells, pooling contributes 9–41 pp of E2E loss (median 20 pp), while Col-Bandit's elimination contributes 0.8–4.0 pp (median 2.0 pp). Col-Bandit's fidelity against the pooled-corpus exhaustive top-5 holds ≥ 0.948 on every cell, with coverage in the deployed 14–21% range and 5.3–7.9× wall-clock speedup vs Ward-Full on the pooled corpus.

Table 9: **Per-dataset Efficiency Analysis: nDCG@K (REAL-MM-RAG).** Coverage budget required to achieve **90%** (white) and **95%** (gray) of Full-MaxSim's **NDCG@K** per corpus, on the Granite-Vision-Embedding (GVE) multimodal benchmarks. **Mean (std)** averages across the four corpora; **Savings** is $100\%/$**Mean**.

Task Domain	**Multimodal (REAL-MM-RAG, GVE)**								**Average**			
Method	**FinSlides**		**FinReport**		**TechSlides**		**TechReport**		*Mean (std)*		*Savings (vs. Full)*	
$K = 5$												
Doc-Uniform	50%	71%	41%	71%	21%	25%	25%	50%	34% (11.7)	54% (18.6)	2.92×	1.84×
Ball-carving	28%	44%	18%	30%	16%	20%	15%	19%	19% (5.3)	28% (9.8)	5.22×	3.54×
Col-Bandit (Ours)	**10%**	**19%**	**8%**	**12%**	**8%**	**9%**	**12%**	**23%**	**10% (1.5)**	**15% (5.7)**	**10.44×**	**6.46×**
$K = 10$												
Doc-Uniform	50%	71%	31%	50%	16%	25%	25%	50%	31% (12.4)	49% (16.0)	3.28×	2.04×
Ball-carving	28%	44%	18%	30%	16%	16%	15%	19%	19% (5.3)	27% (10.9)	5.22×	3.68×
Col-Bandit (Ours)	**11%**	**21%**	**9%**	**13%**	**9%**	**9%**	**14%**	**14%**	**10% (2.0)**	**14% (4.4)**	**9.53×**	**7.07×**
$K = 50$												
Doc-Uniform	41%	71%	25%	50%	16%	25%	25%	50%	27% (8.8)	49% (16.0)	3.72×	2.04×
Ball-carving	28%	44%	18%	30%	16%	16%	15%	19%	19% (5.3)	27% (10.9)	5.22×	3.68×
Col-Bandit (Ours)	**13%**	**18%**	**10%**	**14%**	**11%**	**11%**	**13%**	**16%**	**12% (1.3)**	**15% (2.7)**	**8.60×**	**6.78×**

Table 10: **Per-dataset Efficiency Analysis: Recall@K (BEIR).** We report the coverage budget required to achieve **90%** (white) and **95%** (gray) of Full-MaxSim's **RECALL@K** per corpus. Under **Average**, we report **mean coverage (std)** across datasets, and **Savings (vs. Full)** is the compute reduction factor relative to full reranking (i.e., $100\%/$**Mean**). Quora is shown as *n/a* and excluded from the Average: its end-task metrics are zero across all methods under our harness (a qrels/doc-id mapping issue; Table 4), so per-corpus task-metric budgets are undefined there. Quora retrieval fidelity is unaffected and reported via Overlap@K (Table 6).

Task Domain	**Text Retrieval Benchmarks (BEIR)**									**Average**			
Method	**SciDocs**		**Quora**	**NQ**		**HotpotQA**		**ArguAna**		*Mean (std)*		*Savings (vs. Full)*	
$K = 5$													
ColBERTv2													
Doc-Uniform	41%	50%	*n/a*	22%	22%	72%	91%	50%	50%	46% (18.0)	53% (24.6)	2.16×	1.88×
Ball-carving	20%	26%	*n/a*	17%	17%	30%	34%	36%	44%	26% (7.6)	30% (10.0)	3.88×	3.31×
Col-Bandit (Ours)	**14%**	**14%**	*n/a*	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13% (0.4)**	**13% (0.4)**	**7.55×**	**7.55×**
Jina-ColBERT-V2													
Doc-Uniform	100%	100%	*n/a*	*n/a*		50%	50%	31%	50%	60% (29.1)	67% (23.6)	1.66×	1.50×
Ball-carving	18%	23%	*n/a*	*n/a*		20%	20%	19%	24%	19% (0.8)	22% (1.7)	5.26×	4.48×
Col-Bandit (Ours)	**15%**	**24%**	*n/a*	*n/a*		**7%**	**7%**	**7%**	**7%**	**10% (3.8)**	**13% (8.0)**	**10.34×**	**7.89×**
$K = 10$													
ColBERTv2													
Doc-Uniform	22%	25%	*n/a*	16%	22%	72%	81%	25%	72%	34% (22.3)	50% (26.7)	2.96×	2.00×
Ball-carving	26%	29%	*n/a*	12%	12%	23%	30%	36%	36%	24% (8.6)	27% (8.9)	4.12×	3.74×
Col-Bandit (Ours)	**15%**	**15%**	*n/a*	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**14% (0.9)**	**14% (0.9)**	**7.41×**	**7.41×**
Jina-ColBERT-V2													
Doc-Uniform	50%	91%	*n/a*	*n/a*		31%	50%	20%	50%	34% (12.4)	64% (19.3)	2.97×	1.57×
Ball-carving	15%	18%	*n/a*	*n/a*		20%	20%	19%	19%	18% (2.2)	19% (0.8)	5.56×	5.26×
Col-Bandit (Ours)	**22%**	**60%**	*n/a*	*n/a*		**34%**	**34%**	**7%**	**7%**	**21% (11.0)**	**34% (21.6)**	**4.76×**	**2.97×**
$K = 50$													
ColBERTv2													
Doc-Uniform	16%	41%	*n/a*	12%	16%	31%	72%	25%	25%	21% (7.4)	38% (21.3)	4.76×	2.60×
Ball-carving	20%	26%	*n/a*	12%	12%	23%	23%	26%	26%	20% (5.2)	22% (5.8)	4.94×	4.60×
Col-Bandit (Ours)	**14%**	**17%**	*n/a*	**13%**	**13%**	**13%**	**14%**	**14%**	**14%**	**14% (0.5)**	**14% (1.5)**	**7.41×**	**6.90×**
Jina-ColBERT-V2													
Doc-Uniform	70%	91%	*n/a*	*n/a*		25%	31%	20%	31%	38% (22.5)	51% (28.3)	2.61×	1.96×
Ball-carving	23%	23%	*n/a*	*n/a*		15%	15%	19%	19%	19% (3.3)	19% (3.3)	5.26×	5.26×
Col-Bandit (Ours)	**16%**	**27%**	*n/a*	*n/a*		**10%**	**12%**	**9%**	**10%**	**12% (3.1)**	**16% (7.6)**	**8.57×**	**6.12×**

Table 11: **Per-dataset Efficiency Analysis: Recall@K (REAL-MM-RAG).** Coverage budget required to achieve **90%** (white) and **95%** (gray) of Full-MaxSim's **Recall@K** per corpus, on the Granite-Vision-Embedding (GVE) multimodal benchmarks. **Mean (std)** averages across the four corpora; **Savings** is $100\%/$**Mean**.

Task Domain	**Multimodal (REAL-MM-RAG, GVE)**								**Average**			
Method	**FinSlides**		**FinReport**		**TechSlides**		**TechReport**		*Mean (std)*		*Savings (vs. Full)*	
$K = 5$												
Doc-Uniform	41%	50%	25%	50%	16%	21%	21%	31%	26% (9.3)	38% (12.6)	3.90×	2.64×
Ball-carving	21%	44%	18%	23%	16%	16%	15%	15%	17% (2.4)	24% (11.5)	5.74×	4.11×
Col-Bandit (Ours)	**19%**	**19%**	**8%**	**12%**	**8%**	**9%**	**12%**	**23%**	**12% (4.2)**	**15% (5.7)**	**8.43**×	**6.46**×
$K = 10$												
Doc-Uniform	21%	41%	25%	50%	16%	21%	21%	31%	21% (3.4)	36% (10.9)	4.84×	2.82×
Ball-carving	16%	21%	18%	18%	16%	16%	15%	15%	16% (1.0)	17% (2.4)	6.18×	5.74×
Col-Bandit (Ours)	**21%**	**21%**	**9%**	**13%**	**9%**	**9%**	**14%**	**26%**	**13% (5.0)**	**17% (6.8)**	**7.62**×	**5.79**×
$K = 50$												
Doc-Uniform	11%	16%	11%	16%	11%	16%	11%	25%	11% (0.1)	18% (4.2)	9.33×	5.49×
Ball-carving	12%	16%	14%	14%	16%	16%	15%	15%	14% (1.3)	15% (0.9)	7.03×	6.59×
Col-Bandit (Ours)	**13%**	**18%**	**11%**	**14%**	**11%**	**11%**	**13%**	**16%**	**12% (1.0)**	**15% (2.7)**	**8.38**×	**6.78**×

Table 12: **Per-dataset Efficiency Analysis: MRR@K (BEIR).** We report the coverage budget required to achieve **90%** (white) and **95%** (gray) of Full-MaxSim's **MRR@K** per corpus. Under **Average**, we report **mean coverage (std)** across datasets, and **Savings (vs. Full)** is the compute reduction factor relative to full reranking (i.e., $100\%/$**Mean**). Quora is shown as *n/a* and excluded from the Average: its end-task metrics are zero across all methods under our harness (a qrels/doc-id mapping issue; Table 4), so per-corpus task-metric budgets are undefined there. Quora retrieval fidelity is unaffected and reported via Overlap@K (Table 6).

Task Domain	**Text Retrieval Benchmarks (BEIR)**									**Average**			
Method	**SciDocs**		**Quora**	**NQ**		**HotpotQA**		**ArguAna**		*Mean (std)*		*Savings (vs. Full)*	
$K = 5$													
ColBERTv2													
Doc-Uniform	22%	72%	*n/a*	22%	25%	50%	50%	41%	50%	34% (12.2)	49% (16.6)	2.96×	2.03×
Ball-carving	20%	20%	*n/a*	17%	17%	23%	30%	26%	36%	22% (3.4)	26% (7.6)	4.65×	3.88×
Col-Bandit (Ours)	**13%**	**13%**	*n/a*	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13% (0.0)**	**13% (0.0)**	**7.69**×	**7.69**×
Jina-ColBERT-V2													
Doc-Uniform	91%	100%	*n/a*	*n/a*		50%	70%	50%	50%	64% (19.3)	73% (20.5)	1.57×	1.36×
Ball-carving	15%	18%	*n/a*	*n/a*		20%	26%	19%	24%	18% (2.2)	23% (3.4)	5.56×	4.41×
Col-Bandit (Ours)	**10%**	**10%**	*n/a*	*n/a*		**7%**	**7%**	**7%**	**7%**	**8% (1.4)**	**8% (1.4)**	**12.50**×	**12.50**×
$K = 10$													
ColBERTv2													
Doc-Uniform	22%	72%	*n/a*	22%	50%	50%	50%	41%	50%	34% (12.2)	56% (9.5)	2.96×	1.80×
Ball-carving	20%	20%	*n/a*	12%	17%	23%	30%	26%	36%	20% (5.2)	26% (7.6)	4.94×	3.88×
Col-Bandit (Ours)	**13%**	**13%**	*n/a*	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13% (0.0)**	**13% (0.0)**	**7.69**×	**7.69**×
Jina-ColBERT-V2													
Doc-Uniform	81%	100%	*n/a*	*n/a*		50%	70%	50%	70%	60% (14.6)	80% (14.1)	1.66×	1.25×
Ball-carving	15%	18%	*n/a*	*n/a*		20%	20%	19%	24%	18% (2.2)	21% (2.5)	5.56×	4.84×
Col-Bandit (Ours)	**22%**	**22%**	*n/a*	*n/a*		**7%**	**7%**	**7%**	**7%**	**12% (7.1)**	**12% (7.1)**	**8.33**×	**8.33**×
$K = 50$													
ColBERTv2													
Doc-Uniform	22%	81%	*n/a*	22%	25%	50%	50%	41%	50%	34% (12.2)	52% (19.9)	2.96×	1.94×
Ball-carving	20%	20%	*n/a*	12%	17%	23%	30%	26%	26%	20% (5.2)	23% (5.1)	4.94×	4.30×
Col-Bandit (Ours)	**13%**	**13%**	*n/a*	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13% (0.0)**	**13% (0.0)**	**7.69**×	**7.69**×
Jina-ColBERT-V2													
Doc-Uniform	81%	100%	*n/a*	*n/a*		50%	70%	50%	70%	60% (14.6)	80% (14.1)	1.66×	1.25×
Ball-carving	15%	18%	*n/a*	*n/a*		20%	20%	24%	24%	20% (3.7)	21% (2.5)	5.08×	4.84×
Col-Bandit (Ours)	**10%**	**13%**	*n/a*	*n/a*		**10%**	**10%**	**8%**	**9%**	**9% (0.9)**	**11% (1.7)**	**10.71**×	**9.38**×

Table 13: **Per-dataset Efficiency Analysis: MRR@K (REAL-MM-RAG).** Coverage budget required to achieve **90%** (white) and **95%** (gray) of Full-MaxSim's **MRR@K** per corpus, on the Granite-Vision-Embedding (GVE) multimodal benchmarks. **Mean (std)** averages across the four corpora; **Savings** is $100\%/$**Mean**.

Task Domain	**Multimodal (REAL-MM-RAG, GVE)**								**Average**			
Method	**FinSlides**		**FinReport**		**TechSlides**		**TechReport**		*Mean (std)*		*Savings (vs. Full)*	
$K = 5$												
Doc-Uniform	71%	71%	41%	71%	21%	41%	31%	50%	41% (18.7)	58% (13.0)	2.46×	1.72×
Ball-carving	28%	44%	18%	30%	16%	20%	15%	19%	19% (5.3)	28% (9.8)	5.22×	3.54×
Col-Bandit (Ours)	**10%**	**19%**	**8%**	**12%**	**8%**	**9%**	**12%**	**23%**	**10% (1.5)**	**15% (5.7)**	**10.44×**	**6.46×**
$K = 10$												
Doc-Uniform	71%	71%	41%	71%	21%	25%	31%	50%	41% (18.7)	54% (18.6)	2.46×	1.84×
Ball-carving	28%	44%	18%	30%	16%	20%	15%	19%	19% (5.3)	28% (9.8)	5.22×	3.54×
Col-Bandit (Ours)	**11%**	**21%**	**9%**	**13%**	**9%**	**9%**	**14%**	**14%**	**10% (2.0)**	**14% (4.5)**	**9.53×**	**7.09×**
$K = 50$												
Doc-Uniform	71%	71%	41%	71%	21%	25%	31%	50%	41% (18.7)	54% (18.6)	2.46×	1.84×
Ball-carving	28%	44%	18%	30%	16%	20%	15%	19%	19% (5.3)	28% (9.8)	5.22×	3.54×
Col-Bandit (Ours)	**13%**	**18%**	**9%**	**11%**	**11%**	**11%**	**11%**	**16%**	**11% (1.3)**	**14% (3.2)**	**9.08×**	**7.11×**

Table 14: Retrieval effectiveness at different coverage levels averaged across all nine corpora: *REAL-MM-RAG* (Fin. Reports, Fin. Slides, Tech. Reports, Tech. Slides; Granite Vision Embedding) and *BEIR* (ArguAna, Quora, SciDocs, NQ-2.68M, HotpotQA-500 K; ColBERTv2). Each row is the corpus-mean of the metric (qrel-filtered queries on Quora-Full and HotpotQA-500 K to keep at-least-one-relevant-doc-in-subset; see Appendix A.2). Full reranking at 100% coverage serves as the reference.

Method	Coverage	Recall@1	nDCG@1	MRR@1
Full ColBERT	100%	0.41	0.51	0.51
Col-Bandit	20%	0.40	0.50	0.50
Col-Bandit	40%	0.41	0.50	0.50
Ball-carving	20%	0.33	0.42	0.42
Doc-Uniform	20%	0.23	0.28	0.28
Ball-carving	40%	0.37	0.47	0.47
Doc-Uniform	40%	0.31	0.38	0.38
Relative Retention at 20% Coverage (vs. Full ColBERT)				
Col-Bandit	–	98.9%	98.7%	98.7%
Ball-carving	–	81.0%	82.1%	82.1%
Doc-Uniform	–	55.9%	55.6%	55.6%
Relative Retention at 40% Coverage (vs. Full ColBERT)				
Col-Bandit	–	99.1%	98.9%	98.9%
Ball-carving	–	90.8%	91.9%	91.9%
Doc-Uniform	–	74.9%	74.6%	74.6%

Table 15: **Per-corpus retrieval effectiveness on the four REAL-MM-RAG sub-corpora** (Granite Vision Embedding). Full ColBERT at 100% coverage is the reference; for each method we report Recall@5 / nDCG@5 / MRR@5 at two operating points ($\sim 20\%$ and $\sim 40\%$ coverage). At $\sim 20\%$ coverage Col-Bandit retains $\geq 97.8\%$ of Full ColBERT on every metric and corpus, versus 78–95% for the baselines.

Method	Coverage	Recall@5	nDCG@5	MRR@5
Financial Reports				
Full ColBERT	100%	0.91	0.76	0.70
Col-Bandit	23%	0.91	0.76	0.70
Ball-carving	18%	0.82	0.69	0.64
Doc-Uniform	21%	0.80	0.64	0.59
Col-Bandit	39%	0.91	0.76	0.70
Ball-carving	44%	0.92	0.77	0.72
Doc-Uniform	41%	0.82	0.68	0.64
Financial Slides				
Full ColBERT	100%	0.93	0.77	0.72
Col-Bandit	19%	0.91	0.76	0.72
Ball-carving	21%	0.88	0.68	0.61
Doc-Uniform	21%	0.78	0.61	0.55
Col-Bandit	44%	0.93	0.77	0.72
Ball-carving	44%	0.90	0.74	0.69
Doc-Uniform	41%	0.84	0.67	0.61
Technical Reports				
Full ColBERT	100%	0.94	0.86	0.83
Col-Bandit	23%	0.92	0.85	0.82
Ball-carving	19%	0.93	0.84	0.81
Doc-Uniform	21%	0.85	0.72	0.68
Col-Bandit	36%	0.93	0.86	0.83
Ball-carving	34%	0.96	0.86	0.82
Doc-Uniform	41%	0.92	0.81	0.77
Technical Slides				
Full ColBERT	100%	1.00	0.95	0.93
Col-Bandit	22%	1.00	0.95	0.93
Ball-carving	20%	0.98	0.92	0.89
Doc-Uniform	21%	0.96	0.88	0.85
Col-Bandit	36%	1.00	0.95	0.93
Ball-carving	47%	1.00	0.94	0.92
Doc-Uniform	41%	0.99	0.93	0.91

Table 16: **Col-Bandit per-stage runtime breakdown on CPU-S at K=5, 1 thread.** Deployed knob (α_{ef}=0.2, M=5). The kernel column is the MaxSim micro-kernel (kernel internals in Appendix D.2); Elim and Rescore together stay under 4% of total runtime on every corpus. This is an independent single-thread profiling run, so absolute ms differ slightly from the wall-clock means of Table 2; the takeaway is the stage *proportions*, not the absolute latencies.

Dataset	Cov	Kernel (ms)	Elim (ms)	Rescore (ms)	Total (ms)	Kern %	Bookkeep %
ArguAna	14.7%	36.78	0.318	0.246	37.35	98.5%	1.51%
TechSlides	21.0%	45.87	0.124	1.661	47.66	96.3%	3.75%
TechReport	21.9%	46.50	0.126	1.753	48.37	96.1%	3.88%
FinSlides	17.7%	54.83	0.156	1.929	56.92	96.3%	3.66%
FinReport	23.0%	70.89	0.182	1.936	73.00	97.1%	2.90%
SciDocs	14.3%	81.38	0.616	0.188	82.18	99.0%	0.98%
Quora	12.6%	360.62	9.143	0.020	369.78	97.5%	2.48%
NQ-2.68M	12.6%	4735.23	48.310	0.097	4783.64	99.0%	1.01%

Table 17: **Wall-clock CPU benchmarks on CPU-S, multi-thread breakdown** (AMD EPYC 7763, AVX2; α_{ef}=0.2, M=5, δ=0.01). Per-query latency in ms (1t / 8t). Single-thread summary in the main paper (Table 2).

		K=5					K=100				
Dataset	N	**Ov@5**	**Cov**	**Full (ms)** 1t / 8t	**CB (ms)** 1t / 8t	**Sp.** 1t / 8t	**Ov@100**	**Cov**	**Full (ms)** 1t / 8t	**CB (ms)** 1t / 8t	**Sp.** 1t / 8t
ArguAna	8.7 K	0.98	14%	211 / 29	**33 / 7**	6.4× / 4.2×	0.97	20%	210 / 32	**50 / 10**	4.2× / 3.0×
SciDocs	25.7 K	0.96	14%	513 / 96	**81 / 18**	6.3× / 5.3×	0.93	18%	515 / 96	**108 / 22**	4.8× / 4.4×
HotpotQA	500 K	0.93	13%	5,568 / 728	**757 / 128**	7.4× / 5.7×	0.93	14%	5,153 / 809	**825 / 130**	6.2× / 6.2×
Quora	522.9 K	0.99	13%	1,778 / 242	**381 / 99**	4.7× / 2.4×	0.97	13%	1,806 / 260	**393 / 93**	4.6× / 2.8×
NQ	2.68 M	0.98	13%	37,283 / 4,459	**4,222 / 714**	8.8× / 6.2×	0.97	13%	37,224 / 5,731	**4,384 / 769**	8.5× / 7.5×
MM	8.6 K	0.97	21%	527 / 76	**65 / 20**	8.1× / 3.8×	0.95	47%	523 / 73	**164 / 29**	3.2× / 2.5×
Mean	—	—	—	—	—	**7.0× / 4.6×**	—	—	—	—	**5.3× / 4.4×**

Table 18: **Cross-platform benchmark on CPU-M1, multi-thread breakdown** (Apple M1 Max, NEON; deployed knob). Per-query latency in ms (1t / 8t); vs. m-cpu (Clavié and Lee, 2025) is 1t only. Single-thread summary in the main paper (Table 3).

		K=5						K=100					
Dataset	N	**Ov@5**	**Cov**	**Full (ms)** 1t / 8t	**CB (ms)** 1t / 8t	**Sp. vs Full** 1t / 8t	**vs m-cpu** 1t	**Ov@100**	**Cov**	**Full (ms)** 1t / 8t	**CB (ms)** 1t / 8t	**Sp. vs Full** 1t / 8t	**vs m-cpu** 1t
ArguAna	8.7 K	0.95	14%	284 / 44	**23 / 6**	12.3× / 7.3×	4.6×	0.96	20%	289 / 46	**36 / 8**	8.0× / 5.8×	2.8×
SciDocs	25.7 K	0.92	14%	821 / 120	**68 / 14**	12.1× / 8.6×	6.0×	0.93	18%	832 / 129	**93 / 19**	8.9× / 6.8×	4.4×
MM	8.6 K	0.99	21%	1,636 / 241	**103 / 18**	15.9× / 13.4×	4.0×	0.97	47%	1,657 / 238	**170 / 27**	9.7× / 8.8×	2.4×
HotpotQA	500 K	0.90	13%	7,394 / 1,673	**660 / 140**	11.2× / 11.9×	4.7×	0.94	14%	7,215 / 1,449	**766 / 218**	9.4× / 6.6×	4.3×
Mean	—	—	—	—	—	**12.9× / 10.3×**	**4.8×**	—	—	—	—	**9.0× / 7.0×**	**3.5×**

Table 19: **K-sensitivity** on CPU-S (1 thread, ms) at the deployed knob (α_{ef}=0.2, M=5). Col-Bandit latency grows sub-linearly in K.

	ArguAna		**SciDocs**		**HotpotQA**	
K	Full	CB	Full	CB	Full	CB
5	214	**34**	513	**80**	4,885	**742**
100	223	**47**	551	**105**	5,180	**811**
500	286	**121**	541	**156**	5,171	**911**
1000	243	**154**	585	**217**	5,447	**991**

Table 20: **Operating-point guidance for** α_{ef} (the single Pareto knob). Coverage and Overlap@5 *ranges* across four BEIR text corpora at K=5, M=5, 1 thread. Smaller α_{ef} cuts more aggressively (lower coverage, lower fidelity); α_{ef}=1 is the δ-PAC corner of Theorem 1.

α_{ef}	**Regime**	**Cov**	**Ov@5**
0.10	Aggressive	12.5–12.9%	0.76–0.91
0.20	**Deployed**	**12.6–14.4%**	**0.92–0.99**
0.30	Conservative	12.9–17.4%	0.98–1.00
1.00	Certified	27.7–58.9%	1.00

Table 21: **Commodity CPU Col-Bandit vs. A100 80GB GPU Full-MaxSim** at K=5, α_{ef}=0.2. Per-query latency (ms): GPU runs PyTorch dense Full-MaxSim, CPU-S runs 16-thread Col-Bandit. CPU-S matches GPU within 1.06–1.41× on $N \geq 500$ K text corpora.

Dataset	N	GPU Full (ms)	CPU-S CB 16t (ms)	**Ov@5**
ArguAna	8.7 K	1.9	7.0	0.98
SciDocs	25.7 K	5.4	11.5	0.96
HotpotQA	500 K	65.9	92.7	0.92
Quora	522.9 K	62.6	66.6	0.99
NQ-2.68M	2.68 M	**OOM**	472	0.98

Table 22: Col-Bandit with α_{ef}=0.2 applied to Jina-ColBERT-v2 at two Matryoshka projection sizes (d=128 and d=64). Coverage, overlap, and nDCG are nearly identical across dimensions, confirming that dimension reduction and CB-NK's cell-skipping are orthogonal compression axes that compose without loss.

Dataset	**Dim**	**Cov%**	**Ov@5**	**Ov@100**	**nDCG@5**	**nDCG@100**	**nDCG@5 ret.**
ArguAna	128	24.8	0.996	0.987	0.2858	0.3986	100.0%
	64	25.2	0.997	0.986	0.2922	0.3942	100.0%
SciDocs	128	42.0	0.953	0.921	0.1491	0.2550	95.2%
	64	41.7	0.947	0.923	0.1437	0.2448	93.5%
HotpotQA-500K	128	53.0	0.963	0.950	0.2464	0.2557	100.0%
	64	54.6	0.975	0.962	0.2396	0.2461	99.2%
Quora-Full	128	35.0	0.983	0.953	0.5601	0.5684	100.0%
	64	29.1	0.987	0.954	0.5526	0.5602	100.0%

nDCG@5 ret. = CB-NK nDCG@5 / full-MaxSim nDCG@5 at the same dimension. Full-MaxSim references (d=128): ArguAna 0.2858, SciDocs 0.1567, HotpotQA-500K 0.2464, Quora-Full 0.5601. Full-MaxSim references (d=64): ArguAna 0.2922, SciDocs 0.1536, HotpotQA-500K 0.2416, Quora-Full 0.5526. nDCG@5 can equal the full reference even at Overlap@5 < 1 when the displaced top-5 documents are non-relevant (common on single-relevant corpora such as ArguAna), so 100% retention does not imply an identical top-5 set.

Table 23: **Ward token-axis pooling composes with Col-Bandit.** Per-query latency on CPU-S at the deployed knob (α_{ef}=0.2, K=5). *Fid Ov@5* is overlap with Full-MaxSim on the *pooled* corpus (isolates Col-Bandit's elimination loss); *E2E Ov@5* is overlap with Full-MaxSim on the *unpooled* corpus (combined effect). The end-to-end loss decomposes additively: pooling-induced loss (gap between unpooled Full and the Ward-Full row) and Col-Bandit's elimination cost (gap between the Ward-Full row and the +Col-Bandit row). Pooling is per-document on actual (non-padded) length; pooling dominates by $\sim$10×.

Corpus	**Method**	**Cov (%)**	**Fid Ov@5**	**E2E Ov@5**	**Lat 1t**	**Lat 8t**
SciDocs (text)	Ward 2× Full	100	—	0.912	376	46
	+ Col-Bandit (2×)	14.4	0.952	0.872	**58.8**	**9.4**
	Ward 4× Full	100	—	0.788	211	23
	+ Col-Bandit (4×)	14.2	0.948	0.760	**39.5**	**6.6**
	Ward 8× Full	100	—	0.592	154	14
	+ Col-Bandit (8×)	14.0	0.960	0.572	**25.1**	**4.9**
TechSlides (MM)	Ward 2× Full	100	—	0.884	262	48
	+ Col-Bandit (2×)	20.9	0.968	0.864	**33.3**	**10.7**
	Ward 4× Full	100	—	0.816	137	27
	+ Col-Bandit (4×)	21.1	0.980	0.800	**19.7**	**6.3**
	Ward 8× Full	100	—	0.672	72	14
	+ Col-Bandit (8×)	20.9	0.980	0.664	**11.9**	**3.5**

C Theoretical Validity in Uniform-Sampling Mode (Special Case)

Objective. We seek an algorithm that adaptively reveals entries of H and returns $\widehat{\mathcal{T}}_K$ satisfying $\mathbb{P}(\widehat{\mathcal{T}}_K = \mathcal{T}_K^\star) \geq 1-\delta$ for user-defined $\delta \in (0,1)$, while minimising the expected coverage $\mathbb{E}[\gamma(\Omega)]$ at termination.

Theorem 1 (δ-PAC correctness at α_{ef}=1, conditional on Eq. 7)**.** *For any $\delta \in (0,1)$ and α_{ef}=1, under uniform-without-replacement reveals within each row, and* provided *the radius of Eq. 7 is a valid per-row $(1-\delta_{i,n})$ confidence radius (Remark 2 discusses the c=1 and omitted-$O(1/n)$ simplifications under which we deploy it), Algorithm 1 returns a set $\widehat{\mathcal{T}}_K$ satisfying*

$$\mathbb{P}\Big(\widehat{\mathcal{T}}_K \;=\; \mathcal{T}_K^\star\Big) \;\geq\; 1-\delta. \tag{12}$$

Remark 2 (Scope of the guarantee). Theorem 1 holds for the decision radius of Eq. 7 *as instantiated* at α_{ef}=1: the Bernstein–Serfling pre-factor is set to c=1 and the standard $O(1/n)$ lower-order term is omitted. These simplifications are folded into the radius rather than proved conservative. Where the omitted bias is largest (small n_i) the deterministic hard bounds $[LB_i^{\text{hard}}, UB_i^{\text{hard}}]$ tend empirically to be the binding constraint (Appendix A; an observation, not a conservativeness proof), and the α_{ef}=1 corner empirically attains Overlap@5=1.00 on every text corpus (Appendix B.3), consistent with δ=0.01. The theorem also concerns the *exact-precision* reveal model: the deployed CB-NK kernel makes elimination decisions on int8-quantised estimates (Appendix D.2), a further empirical relaxation for which we make no PAC claim. We therefore describe α_{ef}=1 as δ-PAC *under the radius of Eq. 7 in the exact-precision model*, and treat the deployed α_{ef}=0.2 (with int8 elimination) as a calibrated relaxation (Limitations).

We state a special case in which the simplified empirical Bernstein–Serfling-style radius used in Eq. 7 is δ-valid when $\alpha_{\text{ef}} = 1$. Concretely, a single permutation π of $[T]$ is drawn uniformly and *shared* across all surviving documents; revealing the first n_i entries of π for document i makes $\mathcal{O}_i$ a uniform-without-replacement size-n_i subset of $[T]$ *marginally* for each row. The per-row inequality below depends only on this marginal, so the cross-row coupling induced by the shared π is immaterial (it is handled by the union-bound argument that follows).

Fix a document i and let $\mathcal{O}_i$ be the set of revealed token indices with $n_i = |\mathcal{O}_i|$. Define the row mean and sum

$$\mu_i \triangleq \frac{1}{T}\sum_{t=1}^{T} H_{i,t}, \qquad S_i \triangleq \sum_{t=1}^{T} H_{i,t} = T\mu_i,$$

and the empirical mean/standard deviation over the revealed entries

$$\widehat{\mu}_i = \frac{1}{n_i}\sum_{t\in\mathcal{O}_i} H_{i,t}, \qquad \widehat{\sigma}_i^2 = \frac{1}{n_i-1}\sum_{t\in\mathcal{O}_i}(H_{i,t}-\widehat{\mu}_i)^2.$$

Under uniform-without-replacement sampling within the row and bounded support $H_{i,t} \in [a,b]$, an empirical Bernstein–Serfling inequality (Bardenet and Maillard, 2015) implies that, for any fixed (i,n),

$$\Pr\left(|S_i - T\widehat{\mu}_i| \;\leq\; T\widehat{\sigma}_i\sqrt{\frac{2\log(c/\delta_{i,n})}{n}}\sqrt{\rho_n}\right) \;\geq\; 1-\delta_{i,n}.$$

To obtain a time-uniform statement over all documents and all sample sizes, set $\delta_{i,n} = \delta/(NT)$ and union bound over $i \in [N]$ and $n \in [T]$. The single shared permutation π couples the reveals across rows, but this does not affect the argument: the per-row inequality requires only that each row's revealed set be *marginally* a uniform-without-replacement sample of its T tokens, which holds under a uniformly drawn π, and Boole's inequality holds for arbitrarily dependent events, so the union over rows remains valid regardless of the cross-row coupling. Therefore, with probability at least $1-\delta$, simultaneously for all i and all n,

$$S_i \in \Big[T\widehat{\mu}_i \pm r_i^{\text{th}}(n)\Big], \quad r_i^{\text{th}}(n) \triangleq T\widehat{\sigma}_i\sqrt{\frac{2\log(cNT/\delta)}{n}}\sqrt{\rho_n}.$$

In this uniform-within-row mode, choosing $\alpha_{\text{ef}} = 1$ in Eq. 7 recovers the above theoretical form (up to the constant c), justifying its use as a δ-valid decision radius.

Proof sketch of Theorem 1. We give the high-level argument for the PAC-valid setting (α_{ef}=1, uniform-without-replacement reveals each round).

(1) Per-cell concentration. Each MaxSim cell $H_{i,t} \in [a, b]$ is bounded, and within a row the unrevealed entries form a finite population of size $|\mathcal{U}_i|$ sampled without replacement. The empirical Bernstein–Serfling inequality of Bardenet and Maillard (2015) (Theorem 4.3) therefore yields, for any fixed document i and reveal count n_i, a high-probability two-sided radius for the partial-sum estimator $T\widehat{\mu}_i$ around the row-sum S_i.

(2) Per-round event. At round r, define A_r as the event that, simultaneously for every surviving document $i \in \mathcal{A}_r$, the true score satisfies $\text{LCB}_i \leq S_i \leq \text{UCB}_i$. With $\alpha_{\text{ef}} = 1$ the radius in Eq. 7 matches the Bardenet–Maillard form, so $\mathbb{P}[A_r] \geq 1 - \delta_r$ when the per-round budget δ_r is plugged into the confidence parameter.

(3) Time-uniform union bound. Allocating a per-cell budget $\delta/(NT)$ and union-bounding over all N documents and T reveal counts makes every $[\text{LCB}_i, \text{UCB}_i]$ valid *simultaneously* for all (i, n). Because this holds uniformly over reveal counts, it also holds at the data-dependent stopping time, so the elimination loop needs no separate optional-stopping argument and $\mathbb{P}[\bigcap_r A_r] \geq 1 - \delta$.

(4) Elimination correctness. On $\bigcap_r A_r$, every surviving document obeys $\text{LCB}_i \leq S_i \leq \text{UCB}_i$. If at round r document d has $\text{UCB}_d < \tau_r$, where τ_r is the K-th largest LCB_i over $\mathcal{A}_r$, then at least K documents j satisfy $\text{LCB}_j \geq \tau_r > \text{UCB}_d$; on $\bigcap_r A_r$ each obeys $S_j \geq \text{LCB}_j > \text{UCB}_d \geq S_d$, so d is outranked by $\geq K$ documents, hence $d \notin \mathcal{T}_K^\star$ and removing it from $\mathcal{A}_{r+1}$ is safe. Following the Successive Elimination argument of Kalyanakrishnan et al. (2012); Audibert and Bubeck (2010), repeated application of this rule never discards a true top-K document on the high-probability event.

(5) Termination and rescore. The loop exits with $|\mathcal{A}_r| \leq K{+}M$ or $(r{-}1)B \geq T$. In either case, the K-margin rescore (Algorithm 4) computes the exact score S_i for every survivor using all T tokens, and the final $\arg\text{topK}$ over $\mathcal{A}_r$ is taken on these exact values. On $\bigcap_r A_r$ the survivors include every member of $\mathcal{T}_K^\star$, so the returned set equals $\mathcal{T}_K^\star$.

(6) Combining. Equation (12) follows from $\mathbb{P}[\bigcap_r A_r] \geq 1 - \delta$ and the deterministic implication "$\bigcap_r A_r \Rightarrow \widehat{\mathcal{T}}_K = \mathcal{T}_K^\star$".

Empirical sanity check. At α_{ef}=1 on every text corpus, Algorithm 1 recovers $\widehat{\mathcal{T}}_K = \mathcal{T}_K^\star$ with measured Overlap@5 $\geq 99\%$ (Table 20, α_{ef}=1 row), exceeding the $1-\delta = 0.99$ threshold for $\delta = 0.01$. Increasing coverage drives Overlap@5 toward 1.0 on every corpus, consistent with the radius collapsing as $n_i \to T$.

D Implementation Details and Parameter Selection

D.1 Detailed Algorithm Listing

Algorithm 1 in the main text is condensed for space. Algorithm 4 gives the fully-annotated version, making the per-round bookkeeping (Ω, $\widehat{S}_i$, the explicit batch set $\mathcal{B}_r$) and the two-phase structure (elimination loop followed by exact rescore) explicit.

Walkthrough. *Initialisation (lines 1–3).* All N candidates start in the active set $\mathcal{A}_1$; the observed set Ω is empty and every interval is the trivial support bound $[aT, bT]$ (no cells revealed yet). A single per-query permutation π fixes the reveal order of the T query tokens; it is drawn uniformly with a fixed seed and shared across all documents, which is the structural condition under which Theorem 1 applies (Appendix C).

Elimination loop (lines 4–13). Each round selects the next B tokens of π (line 5) and, in a single vectorised pass over the surviving set (lines 6–9), reveals those B cells for every active document, appends them to Ω, and refreshes the partial-sum estimate $\widehat{S}_i$ and the hybrid interval $[\text{LCB}_i, \text{UCB}_i]$ via Eq. 8. The elimination threshold τ_r is the K-th largest lower bound over the active set (line 10); any document whose *upper* bound is below τ_r cannot belong to the top-K and is permanently dropped (line 11). The loop exits once the active set is small enough for exact completion ($|\mathcal{A}_r| \leq K{+}M$) or all T tokens have been revealed.

Exact rescore (lines 14–16). The $\leq K{+}M$ survivors are rescored on all T query tokens with the high-precision kernel (the same fused call as Full-MaxSim, identical float-add order), so the returned

Algorithm 4 Col-Bandit: batched progressive elimination for top-K identification on the late-interaction matrix (detailed).

Require: Query Q (T tokens), candidate set $\mathcal{D} = \{d_1, \ldots, d_N\}$, target K, margin M, batch B, calibration α_{ef}, confidence δ
Ensure: Estimated top-K set $\widehat{\mathcal{T}}_K \subseteq \mathcal{D}$

$\mathcal{A}_1 \leftarrow [N], \quad \Omega \leftarrow \emptyset, \quad r \leftarrow 1$ ▷ Active set; observed entries; round counter
$[\mathrm{LCB}_i, \mathrm{UCB}_i] \leftarrow [a\,T,\, b\,T]$ for all $i \in [N]$ ▷ Trivial initial interval
$\pi \leftarrow$ random permutation of $[T]$ (uniform, fixed seed; Appendix C)
while $|\mathcal{A}_r| > K + M$ **and** $(r-1)B < T$ **do**
  $\mathcal{B}_r \leftarrow \{\pi[(r-1)B+1], \ldots, \pi[\min(rB, T)]\}$ ▷ Next B tokens of π
  for all $i \in \mathcal{A}_r$ **in parallel** (one SIMD pass) **do**
    Reveal $\{H_{i,t} : t \in \mathcal{B}_r\}$; $\quad \Omega \leftarrow \Omega \cup (\{i\} \times \mathcal{B}_r)$
    Update $\widehat{S}_i$ and $[\mathrm{LCB}_i, \mathrm{UCB}_i]$ via Eq. 8 (radius $\propto \alpha_{\text{ef}}, \delta$)
  end for
  $\tau_r \leftarrow K$-th largest LCB_i over $i \in \mathcal{A}_r$ ▷ Elimination threshold
  $\mathcal{A}_{r+1} \leftarrow \{i \in \mathcal{A}_r : \mathrm{UCB}_i \geq \tau_r\}$ ▷ Permanent elimination
  $r \leftarrow r + 1$
end while
for all $i \in \mathcal{A}_r$ **do** ▷ K-margin rescore on survivors
  $S_i \leftarrow \sum_{t=1}^{T} H_{i,t}$ using high-precision kernel
end for
return $\widehat{\mathcal{T}}_K \leftarrow \arg\mathrm{topK}_{i \in \mathcal{A}_r} S_i$

scores are bit-identical to the exhaustive baseline and the final top-K is exact over the survivor set. The margin M gives the elimination loop slack: borderline documents that the loop would otherwise have to disambiguate with more reveals are instead deferred to this cheap exact rescore.

Complexity. The loop reveals $B|\mathcal{A}_r|$ cells per round and the active set shrinks monotonically, so the total cells touched are $\sum_r B|\mathcal{A}_r| + T\,|\mathcal{A}_{\text{final}}|$, which is far below the NT of exhaustive scoring whenever the score distribution has a clear top-K separation (the regime confirmed empirically in §5). The bookkeeping (interval updates, threshold, elimination) is $O(|\mathcal{A}_r|)$ per round and is dominated by the kernel cost (Appendix D.2, Table 16).

D.2 The NUMKONG C Extension

The Full-MaxSim baseline and Col-Bandit share a single C extension (referred to throughout as our NUMKONG extension), so that the wall-clock comparisons in Table 2 and Table 3 are apples-to-apples by construction: only the reveal schedule differs, not the kernel quality. Our NUMKONG-based Col-Bandit kernel (CB-NK) will be released as open-source software upon publication. It is not publicly available at submission time, to preserve review anonymity.

Target instruction sets. The same C source supports three SIMD back-ends, selected at compile time: (i) x86-64 AVX2 (256-bit, 8 fp32 lanes), used on CPU-S (AMD EPYC 7763) for Table 2; (ii) ARMv8 NEON (128-bit, 4 fp32 lanes), used on CPU-M1 (Apple M1 Max) for the edge-deployment study (Table 3); (iii) x86-64 AVX-512 (512-bit, 16 fp32 lanes), which compiles cleanly but is not benchmarked in this submission. The dot-product micro-kernel is the only platform-specific piece of code; the algorithmic skeleton (bound updates, elimination, K-margin rescore) is shared across all three back-ends.

Memory layout. Documents are stored as a packed flat array in tile-friendly stride order, so that per-document MaxSim accumulation reads sequential cache lines. Padding rows are zero-norm, so they cannot dominate any per-query-token max and do not bias the score.

Compute path. Full-MaxSim runs in fp32 throughout. Col-Bandit (CB-NK) uses an int8-quantised coarse pass for the cheap LCB/UCB updates inside the elimination loop, followed by an aligned fp32 rescore for the $K+M$ surviving documents. The $K+M$ rescore swaps to Full-MaxSim's exact kernel call (same float-add order, same packed input layout), so survivor scores are bit-identical to the exhaustive baseline; this property is what allows Overlap@K to be a clean fidelity measure rather than a quantisation-noise measure.

Multi-threading model. Parallelism is across queries via OpenMP: each query is processed by a single thread, so the 1-thread vs. 8-thread columns in Table 2 (and the corresponding M1 figures) represent a

fixed query batch where threads partition the query set. This avoids intra-query synchronisation overhead and matches the deployment regime where queries arrive concurrently.

Default parameter setting. Unless stated otherwise, all Col-Bandit experiments use the following deployed configuration: $\alpha_{\text{ef}}=0.20$ (the Pareto figures additionally sweep $\alpha_{\text{ef}} \in \{0.01, 0.02, 0.05, 0.10, 0.20, 0.30, 0.40, 0.50, 1.0\}$); round size $B=4$ (number of query tokens revealed per elimination round); rescore margin $M=5$ (extra survivors carried into the final exhaustive-aligned rescore pass); PAC failure probability $\delta=0.01$; empirical Bernstein–Serfling bounds (Bardenet and Maillard, 2015) (§4.1). The reveal order is a Fisher–Yates random permutation of $[T]$ with a fixed seed for reproducibility; at $\alpha_{\text{ef}}=1$ this matches the conditions of Theorem 1.

Hardware platforms. The hardware platforms used in the experiments are: CPU-S (AMD EPYC 7763, AVX2) for Table 2; CPU-M1 (Apple M1 Max, NEON) for Table 3; and GPU (NVIDIA A100 80 GB) for all GPU sweeps and the Pareto figure (Figure 8).

Hardware realization. Algorithm 1's batched structure aligns naturally with modern SIMD execution: each round reveals $B=4$ tokens for all surviving documents in a single vectorized pass. We implement Col-Bandit as a fused C kernel parameterized by the underlying instruction set (AVX2/AVX-512 on x86, NEON on Apple Silicon) using a cache-separated quantized memory layout. The wall-clock results in Section 5.2 use this fused realization on both Intel/AMD x86 and Apple Silicon.

Cache-separated quantized layout. Document token embeddings are stored in two parallel buffers per document: an int8 buffer used for fast partial-sum updates inside the elimination loop, and an aligned fp32 buffer used by the final K-margin rescore. The int8 layout uses percentile-clip quantization (clip to the 99.9th percentile of absolute values, then linear quantize to int8) so that AVX-512 `VPDPBUSD` / NEON `sdot` dot-products run at full throughput while the rescore uses bit-faithful float arithmetic. Because elimination decisions use the int8 estimates while the $K+M$ survivors are rescored in exact fp32, the survivor *scores* are always exact, but quantization can change *which* borderline documents survive to the rescore. Any resulting top-K error is therefore already reflected in our reported Overlap@K, which is measured end-to-end against exact Full-MaxSim, and the margin M absorbs most such borderline cases. Because int8 dot products round differently across SIMD back-ends, the surviving set, and hence Overlap@K, can differ slightly between CPU-S (AVX2) and CPU-M1 (NEON) on the same data.

Register tiling. The inner loop processes a $B=4$ batch of query tokens against 4 document token positions per cycle, which fits in 4 AVX-512 / NEON vector registers and keeps the hot data in L1. The four query tokens of the current $\mathcal{B}_r$ are kept in registers across all surviving documents in a round so that document tokens are streamed in once.

Cross-platform. The same C source compiles unchanged for x86 (AVX2/AVX-512) and ARM (NEON, including Apple Silicon). The only platform-specific code is the dot-product micro-kernel, which is selected at compile time via preprocessor macros. Empirically the NEON port on CPU-M1 matches the AVX2 wall-clock on CPU-S at single-thread on small corpora and exceeds it at larger K.

Cross-ISA comparison against *maxsim-cpu*. The published *maxsim-cpu* (Clavié and Lee, 2025) Rust SIMD MaxSim is tuned primarily for Apple Silicon, so its standing relative to our NUMKONG kernel depends on the instruction set. On CPU-M1 (NEON) it is faster than our Full-MaxSim, yet Col-Bandit is still $4.8\times$ faster than it at single thread (Table 3). On x86 the ordering reverses: on an AMD EPYC Zen-3 server (AVX2, the same instruction set and microarchitecture family as CPU-S), our NUMKONG Full-MaxSim is already $\sim 2.3\times$ faster than *maxsim-cpu*, so Col-Bandit widens to $\sim 11\times$ over it at single thread (ArguAna $11.2\times$, SciDocs $11.5\times$; $K=5$, $\alpha_{\text{ef}}=0.2$, Overlap@$5 \geq 0.91$). The kernel-level acceleration thus holds against a production third-party SIMD baseline on both ISAs; in particular, the AVX2 result shows that Col-Bandit's speedup is not an artifact of a slow Full-MaxSim baseline, since that baseline itself outpaces *maxsim-cpu* on x86. This x86 measurement was taken on an EPYC Zen-3 (AVX2) node rather than the exact CPU-S 7763; our Full-MaxSim there tracked the CPU-S Full timings of Table 2

Table 24: **x86 cross-ISA check vs. *maxsim-cpu*** (single thread, K=5, α_{ef}=0.2, M=5, δ=0.01) on an AMD EPYC Zen-3 (AVX2) node, an ISA-matched proxy for CPU-S. Col-Bandit (CB-NK) is ∼11× faster than *maxsim-cpu*, and our NUMKONG Full-MaxSim is itself ∼2.3× faster than *maxsim-cpu* on this ISA.

Dataset	N	**Cov**	**Ov@5**	**Full (ms)**	**m-cpu (ms)**	**CB (ms)**	**CB vs m-cpu**
ArguAna	8.7 K	14%	0.93	180	404	36	11.2×
SciDocs	25.7 K	14%	0.92	509	1,174	102	11.5×

Table 25: **K-margin rescore ablation** (α_{ef}=0.2, K=5, 1 thread, 50 queries per corpus). Without the rescore (M=0) borderline-eliminated documents leak through; M=5 restores Ov@5 $\geq$ 0.98 on both corpora at a small latency cost. Numbers are not directly comparable to the full-benchmark Ov@5 in Table 2, which uses the complete per-corpus query set.

Dataset	M	Cov	Ov@5	Lat. (ms)
SciDocs	0	18%	0.94	107
	5	18%	0.98	129
	10	19%	0.98	129
HotpotQA	0	14%	0.93	790
	5	14%	0.98	838
	10	14%	0.98	853

(ArguAna 180 vs. 211 ms; SciDocs 509 vs. 513 ms), supporting it as an ISA-matched proxy. Table 24 reports the full breakdown.

D.3 Parameter Selection

Col-Bandit exposes two practical knobs: the calibration parameter α_{ef} and the margin M. In practice, α_{ef} governs the aggressiveness of elimination: smaller values tighten the decision radius and reduce coverage, while larger values are more conservative. The margin M has a small, monotone effect: increasing M from 0 to 5 adds roughly 2 percentage points of Overlap@K at negligible runtime cost, since rescoring $K+M$ documents on all T tokens is a fixed small overhead. We use M=5 by default and select α_{ef} based on a desired quality–coverage trade-off. Table 25 sweeps $M \in \{0, 5, 10\}$ on two representative text corpora (SciDocs, HotpotQA-500 K) at α_{ef}=0.2, K=5, 1 thread.

Calibrated relaxation. The relaxation parameter $\alpha_{\text{ef}} \in (0, 1]$ rescales the Bernstein–Serfling radius of Eq. 7 (the same Bardenet–Maillard radius that appears in the proof of Theorem 1), applied at every elimination step. At α_{ef}=1 the bound is the unshrunk simplified Bernstein–Serfling-style radius and Theorem 1 delivers its conditional δ-PAC guarantee. For α_{ef}<1 the radii shrink, more aggressive eliminations are permitted, and the formal guarantee weakens to a calibrated relaxation: empirically, the rank-recovery error grows smoothly and predictably with the coverage saving. Crucially, α_{ef} is a single continuous knob with an interpretable δ-PAC endpoint at α_{ef}=1, so practitioners can dial in a desired quality–coverage operating point along a one-dimensional Pareto.

E Taxonomy of Efficient Late-Interaction Retrieval

Figure 9 situates Col-Bandit relative to existing late-interaction acceleration methods along two axes: *when* pruning happens (index-time vs. query-time) and *what* is pruned (candidates, tokens, embedding dimensions, or atomic MaxSim cells). Index-time methods (PLAID, MUVERA, token pruning) commit before seeing the query and therefore prune conservatively; query-time methods (Ball-carving, Col-Bandit) can use the actual query, with Col-Bandit, to our knowledge, the first to prune at the atomic MaxSim-cell level.

E.1 Extended Related Work

This expands the condensed discussion in §2.2.

Index-Time Compression & Token Pruning. Approaches like PLAID (Santhanam et al., 2022a), ColBERTv2 (Santhanam et al., 2022b), and MUVERA (Dhulipala et al., 2024) accelerate retrieval via

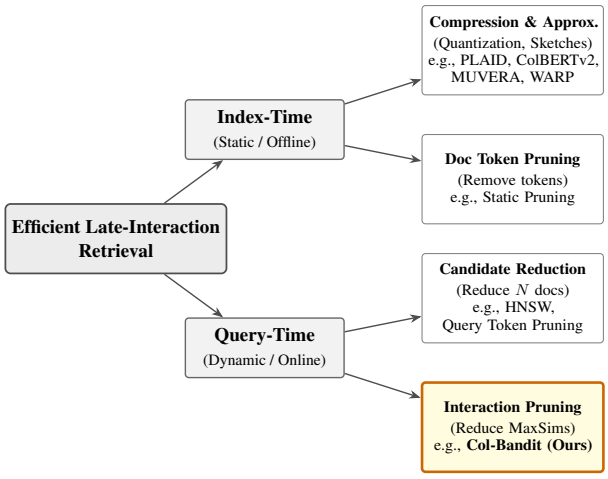

Figure 9: **Taxonomy of efficient late-interaction retrieval.** Methods are classified by *when* they prune (index-time vs. query-time) and *what* they prune. Col-Bandit is the first to dynamically prune the atomic interaction matrix H during query-time scoring.

centroid-based compression, quantization, or fixed-dimensional encodings, improving the practicality of late-interaction methods that were initially constrained by considerable storage requirements. More recently, LEMUR (Jääsaari et al., 2026) reduces multi-vector retrieval to single-vector ANN through a learned per-corpus MLP, exact-MaxSim reranking the top-k' ANN candidates. Additional system and indexing advances such as WARP (Scheerer et al., 2025) further improve scalability and usability. Similarly, token pruning methods (Lassance et al., 2021; Tonellotto and Macdonald, 2021) permanently discard non-informative tokens to reduce the index size (N) or query length (T), including near-lossless vector count reduction (Clavié et al., 2024) and approaches that use a fixed number of representative tokens (MacAvaney et al., 2025). While effective, these methods are fixed at index-time and typically require offline modifications. Col-Bandit is *orthogonal* to them: it operates purely at query-time on standard indices, dynamically pruning the atomic interaction matrix H during scoring. The two layers can be combined: *any* of the above index-time systems can produce the candidate set $\mathcal{D}$ that Col-Bandit reranks.

Efficient Systems & Bound-Based Pruning. System-level optimizations like DESSERT (Engels et al., 2023) use approximate retrieval to speed up candidate generation. In sparse retrieval, algorithms like WAND (Broder et al., 2003) and BMW (Ding and Suel, 2011) use score upper bounds to skip documents. Col-Bandit bridges these concepts, applying bound-based early stopping to *dense late-interaction*. Unlike WAND, which prunes inverted list pointers, we prune atomic MaxSim operations $h(d, t)$ to certify the Top-K set with statistical guarantees.

MaxSim-Level Pruning (Our Approach). To our knowledge, no prior work adaptively *estimates* the exhaustive top-K identity from a partially-revealed MaxSim matrix *within* the exact scoring loop, using the actual query to decide which cells to compute. Existing methods reduce the number of candidates (N) or tokens (T) *before* scoring, and therefore before seeing the query. Col-Bandit frames the scoring process itself as a finite-population top-K identification problem, using the query to drive the elimination, and progressively revealing only the subset of MaxSim entries needed to certify the ranking.

Finite-Population Bandits and Top-k Arm Identification. Our method is inspired by fixed-confidence Top-K Arm Identification (Kalyanakrishnan et al., 2012; Chen et al., 2014), with one structural twist: standard best-arm identification (BAI) estimates the unknown means of *stochastic* arms, whereas a row of our score table H is a *finite* population of T deterministic values sampled *without replacement*; the randomness is only in the reveal order. Fixed-confidence BAI machinery (UCB (Auer et al., 2002), UCB-E (Audibert and Bubeck, 2010), LUCB (Kalyanakrishnan et al., 2012)) gives us the interval-driven reveal policy and stopping criterion, but standard infinite-population bounds are conservative here. We therefore replace the stochastic-arm radius with the empirical Bernstein–Serfling concentration of Bardenet and Maillard (2015), which collapses deterministically as $n_i \to T$ and yields tighter intervals (§4.1). MAB

applications include prompt learning (Shi et al., 2024), LLM evaluation (Zhou et al., 2024), k-NN search (LeJeune et al., 2019).

F Table of Notations

The notations used in the paper are described below.

Table 26: Notations used in the paper.

Symbol	Description
Input	
$Q = \{\mathbf{q}_1, \ldots, \mathbf{q}_T\}$	A query represented as a set of T token embeddings
d	A document from the collection $\mathcal{D}$
$\mathcal{D} = \{d_1, \ldots, d_N\}$	The candidate document set with N documents
T	The number of query tokens
L_d	The number of tokens in document d
l	The embedding dimension ($\mathbf{q}_t, \mathbf{e}_{d,j} \in \mathbb{R}^l$)
K	The number of top documents to identify
Scoring	
$\mathrm{sim}(\cdot,\cdot)$	Similarity function (e.g., cosine similarity); bounded support $[a, b]$
$h(d, t)$	MaxSim score: $\max_{j\in[L_d]} \mathrm{sim}(\mathbf{e}_{d,j}, \mathbf{q}_t)$
$S(d; Q)$	Total late-interaction score: $\sum_{t=1}^{T} h(d, t)$
$\mathcal{T}_K^\star$	The true (exhaustive) Top-K document set
$\widehat{\mathcal{T}}_K$	The returned (estimated) Top-K document set
$\mathrm{Overlap@}K$	Headline fidelity metric: $\lvert\mathcal{T}_K^\star \cap \widehat{\mathcal{T}}_K\rvert/K$ (Eq. 9)
Matrix & Observation	
$\mathbf{H} \in \mathbb{R}^{N\times T}$	The MaxSim matrix with entries $H_{i,t} = h(d_i, t)$
S_i	Total score for document i: $\sum_{t=1}^{T} H_{i,t}$
$\Omega \subseteq [N] \times [T]$	The set of observed (revealed) matrix entries
$\mathcal{O}_i$, $\mathcal{U}_i$	Observed and unobserved token indices for i; $\mathcal{U}_i = [T] \setminus \mathcal{O}_i$
n_i	Number of revealed tokens for document i: $\lvert\mathcal{O}_i\rvert$
$\gamma(\Omega)$	Coverage: fraction of matrix revealed, $\lvert\Omega\rvert/(N \times T)$
π	Per-query reveal permutation of $[T]$ (uniform, fixed seed)
Bounds & Algorithm State	
$[a, b]$	Global support of $H_{i,t}$ (e.g., $[-1, 1]$ for cosine similarity)
$\hat{\mu}_i$	Empirical mean over revealed entries: $\frac{1}{n_i}\sum_{t\in\mathcal{O}_i} H_{i,t}$
$\hat{S}_i$	Partial-sum estimator: $T \cdot \hat{\mu}_i$
$\hat{\sigma}_i$	Empirical standard deviation over $\{H_{i,t}\}_{t\in\mathcal{O}_i}$
$\mathrm{LB}_i^{\mathrm{hard}}, \mathrm{UB}_i^{\mathrm{hard}}$	Deterministic hard bounds (§4.1)
r_i^{eff}	Variance-adaptive decision radius (Eq. 7)
$\mathrm{LCB}_i, \mathrm{UCB}_i$	Hybrid lower/upper confidence bounds (Eq. 8)
$\mathcal{A}_r$	Active set of surviving documents at round r
$\mathcal{B}_r$	Set of B token indices revealed in round r
τ_r	Round-r elimination threshold: K-th largest LCB_i over $\mathcal{A}_r$
Algorithm Parameters	
$\alpha_{\mathrm{ef}} \in (0, 1]$	Calibration knob; $\alpha_{\mathrm{ef}}{=}1$ is the δ-PAC corner (Theorem 1)
M	Rescore margin: $K{+}M$ survivors rescored exactly on all T tokens
B	Round (batch) size: tokens revealed per round
$\delta \in (0, 1)$	PAC error tolerance
ρ_n	Finite-population correction factor (Eq. 7)