COL-BANDIT: QUERY-TIME TOP-K ESTIMATION FOR LATE-INTERACTION RETRIEVAL

Anonymous authors
Paper under double-blind review

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 cast this as *fixed-confidence top-K identification over a finite population* and introduce Col-Bandit, a query-time estimator of the exhaustive-MaxSim top-K: it reveals matrix entries in batches without replacement, maintains a finite-population Bernstein–Serfling confidence interval on each candidate's score, whose radius collapses to zero at full reveal, and permanently drops any document whose upper bound shows it cannot finish in the top-K, computing only the cells needed to separate the top-K. A single relaxation knob $\alpha_{\text{ef}} \in (0, 1]$ tunes the compute–fidelity trade-off. We deploy $\alpha_{\text{ef}}{=}0.2$, while the conservative $\alpha_{\text{ef}}{=}1$ admits a conditional δ-PAC guarantee under the implemented 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 mean single-thread CPU speedups of $7.0\times$ on x86 and $12.9\times$ on ARM, peaking at $\sim 16\times$. A drop-in reranking layer, it needs no retraining or index changes. Our code is released as open-source software (anonymized repository: `https://anonymous.4open.science/r/ColBandit-3DBE`).

1 INTRODUCTION

Multi-vector late-interaction retrievers, such as ColBERT (Khattab & 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; IBM Research Team, 2025; Warner et al., 2025; Nomic Team, 2025; 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 hiring the top-K of N applicants, each scored by the sum of T tests: an efficient manager tests in rounds and drops applicants who cannot reach the top-K even by acing every remaining 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 score revelation*: the per-token MaxSim scores form an $N \times T$ matrix, one row per candidate document and one column per query token, whose row-sums are the final scores. Entries are revealed on demand, and our objective is to estimate the exhaustive scorer's Top-K identity from a partially revealed matrix (never to reconstruct the matrix itself), minimizing computation at 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 operates *inside the scoring stage itself*, an allocation unavailable to

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

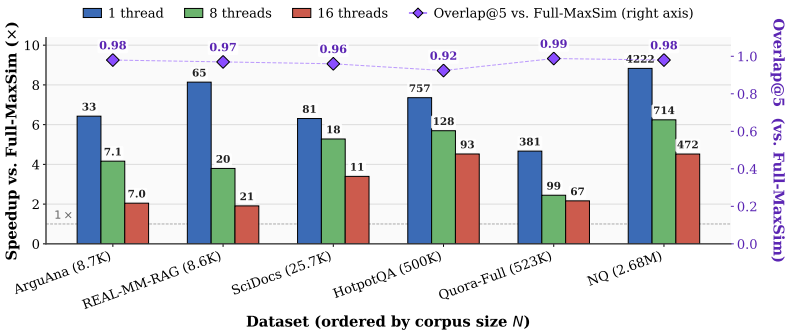

Figure 1: **Wall-clock speedup of Col-Bandit vs. Full-MaxSim across BEIR and REAL-MM-RAG** on CPU-S at the deployed knob (α_{ef}=0.2, M=5, δ=0.01, K=5). Bars: speedup at 1/8/16 threads, labeled with Col-Bandit per-query latency (ms); datasets ordered by corpus size. Purple diamonds (right axis): Overlap@5, ≥ 0.92 on every corpus (HotpotQA-500 K is the floor). The REAL-MM-RAG entry averages the four multimodal corpora (combined N=8.6 K), each benchmarked separately. Other α_{ef} and K: Appendix D.3.

methods that fix their representation or pruning structure before scoring begins (the relevant cells are unknowable before the query arrives). We call the algorithm CB and its kernel implementation on the NUMKONG (Vardanian) C extension CB-NK. Unlike prior acceleration methods that *lossily* compress or distill document representations, Col-Bandit targets the top-K of the exhaustive reference scorer itself (survivors are exactly rescored on all T tokens, and the elimination rule is conditionally correct at α_{ef}=1; Remark 2) and requires **no retraining and no changes to the model or retrieval index** (one offline embedding repack; Appendix F). We formulate the task as *fixed-confidence Top-K identification over a finite population*: each document's score is a sum of T fixed, bounded values revealed on demand, sampled *without replacement*. This departs from classical best-arm identification and bandit-style similarity search (Tiwari et al., 2022) in two ways. First, the population is finite and small (T=32), so we replace the usual sub-Gaussian radius with the empirical Bernstein–Serfling inequality (Bardenet & Maillard, 2015), whose without-replacement radius shrinks to *zero* at full reveal, so the algorithm reaches exactness at finite reveal rather than asymptotically. Second, each revealed cell $h(d,t)$, the entry for document d and token t, is itself a $\max$ over L_d document-token similarities computed at query time, and the reveal budget is allocated *per document* by elimination timing: candidates near the top-K boundary keep receiving reveals; clear losers exit early.

Contributions.

- **Finding.** Identifying the exhaustive top-K does not require the full matrix: at the deployed setting, 13–22% of the MaxSim cells suffice to preserve Overlap@5 ≥ 0.90 on all nine corpora; the $N \times T$ matrix is massively redundant for ranking.
- **Formulation and algorithm.** We cast late-interaction reranking as finite-population Top-K identification, using the query to decide, inside the scoring stage, which cells to compute and which documents to stop scoring, and introduce Col-Bandit, a progressive elimination algorithm with empirical Bernstein–Serfling confidence bounds (Bardenet & Maillard, 2015) and one relaxation knob α_{ef} (conditional δ-PAC guarantee at α_{ef}=1; Eq. 5, Remark 2). The deployed default α_{ef}=0.2 is a calibrated relaxation.
- **Efficient CPU realization.** A fused SIMD kernel, released as open-source software, turns the cell savings into wall-clock: $\mathbf{7.0}\times$ (AMD EPYC, AVX2) and $\mathbf{12.9}\times$ (Apple M1 Max, NEON) mean single-thread speedups at K=5 (Figure 1), with bit-identical survivor scores, no index modifications or retraining, and essentially free bookkeeping (no-elimination control, §5.3). Latency scales *sub-linearly* in K (5.3×/9.0× mean at K=100).
- **Benchmark and guidance.** We measure both compute (coverage) and latency across BEIR and REAL-MM-RAG (up to 2.68 M documents), plus what a practitioner needs: sensitivity to retrieval depth, query length, and design choices; composition with first-stage retrieval engines; and a single calibrated α_{ef} that transfers across all corpora with no per-corpus tuning (§5).

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*: each query token $t \in [T]$ contributes the similarity of its best-matching document token, the per-token MaxSim $h(d,t)$, and these contributions sum to the document's relevance score $S(d;Q)$:

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

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

where $\mathrm{sim}(\cdot,\cdot)$ is a similarity function (typically cosine).[1]

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 relevance 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 fix their representations and pruning structure before the query arrives; at search time some (e.g., PLAID) still use the query to select *candidates*, but the scoring stage that follows evaluates every MaxSim cell of every candidate. Col-Bandit is, to our knowledge, the first method to allocate work adaptively at the atomic MaxSim-cell level *within* that scoring stage. Standard reranking evaluates all $N \times T$ MaxSim values $h(d,t)$ (Eq. 1), 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 G). 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), WARP (Scheerer et al., 2025), TACHIOM (Martinico et al., 2026), and SSR (Guo et al., 2026), accelerate retrieval by quantizing or projecting document representations offline; at search time these systems use the query to generate and prune *candidates* (e.g., PLAID's query–centroid pruning), but their final scoring stage typically evaluates every retained candidate exhaustively. Offline token pruning (Lassance et al., 2021; Clavié et al., 2024; MacAvaney et al., 2025) permanently discards low-utility document or query tokens, committing at index time. Col-Bandit is *orthogonal* and stacks on top, since any such system can produce the candidate set $\mathcal{D}$ that Col-Bandit reranks. The distinction matters for evaluation, too: WARP is a retrieval *engine* (candidate generation and decompression for XTR/PLAID-style indices) and LEMUR *learns* a per-corpus single-vector reduction whose top-k' is then exact-MaxSim reranked, so head-to-head latency comparisons would conflate stage-1 candidate quality with stage-2 scoring efficiency. The meaningful comparison is replacement on identical candidates, which we measure for PLAID, LEMUR, and MUVERA in §5.3.

[1] 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.

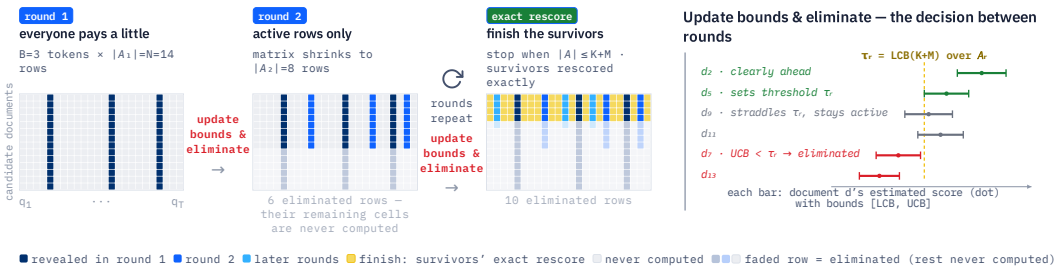

Figure 2: **Col-Bandit computes only the MaxSim cells that decide the top-K.** *Left:* the $N \times T$ MaxSim matrix (one row per candidate document, one column per query token q_t; cell (d, t) is the per-token MaxSim $h(d, t)$ of Eq. 1, a document's score is its row-sum, Eq. 2) as the rounds progress (intermediate rounds elided, ↻). Each round reveals a shared batch of B query tokens for the active documents $\mathcal{A}_r$ only; eliminated documents sink to the bottom, faded; their computed cells were the price of pruning (*pay-to-prune*), and the rest is never computed. When $|\mathcal{A}| \leq K+M$, the survivors are rescored exactly over all T tokens (the small $K+M$ rescore, counted separately). *Right:* the between-round decision: each batch tightens every active candidate's interval $[\mathrm{LCB}_i, \mathrm{UCB}_i]$ (Eq. 6); the $(K+M)$-th best lower bound sets τ_r, and any document with $\mathrm{UCB}_i < \tau_r$ is permanently eliminated (Algorithm 1). At the deployed setting (K=5), Col-Bandit computes 13–22% of the matrix, for mean single-thread CPU scoring speedups of 7.0× (AVX2) and 12.9× (Apple NEON) at Overlap@5 ≥ 0.90 on every tested corpus, using one fused SIMD kernel (int8 argmax candidates, fp32 refinement) whose survivor scores are bit-identical to the exhaustive reference. Illustration is qualitative and not to scale (traced run: Appendix F.2).

Bound-based skipping. In sparse retrieval, WAND (Broder et al., 2003) and BMW (Ding & 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 descends from fixed-confidence pure exploration: LUCB-style Top-K identification (Kalyanakrishnan et al., 2012; Chen et al., 2014), action elimination (Even-Dar et al., 2006; Audibert & Bubeck, 2010), and their application to similarity search in bounded-pull MIPS bandits (Liu et al., 2019) and BanditMIPS (Tiwari et al., 2022), which adaptively subsample coordinates of flat inner products. Three features separate our setting from this line. *(i) Finite population, without replacement:* each MaxSim row is a fixed set of T deterministic values (T=32 here), not a stochastic stream; we sample cells without replacement under the empirical Bernstein–Serfling concentration of Bardenet & Maillard (2015), whose radius *collapses to zero at full reveal*, unlike sub-Gaussian radii (as in BanditMIPS), which never vanish, require a known variance proxy, and pay off only asymptotically in the dimension ($d \sim 10^3$ there vs. our T=32). *(ii) Per-document budgets via elimination timing:* all survivors share one uniform reveal permutation (keeping the without-replacement analysis valid and each round a single SIMD pass); adaptivity comes from per-document stopping, concentrating budget on candidates near the top-K boundary. *(iii) Expensive on-demand entries:* each revealed value is itself a max over L_d token similarities computed at query time, so the unit of cost is the quantity being economized, unlike threshold algorithms over precomputed sorted lists (Fagin et al., 2001).

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 $\alpha_{\mathrm{ef}} = 1$ (conditionally; Theorem 1, Remark 2). We cast this as a fixed-confidence Multi-Armed Bandit (MAB) problem over the sparsely observed, finite-population MaxSim matrix.

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 $\widehat{\mathcal{T}}_K$.
$r \leftarrow 1$; $\mathcal{A}_1 \leftarrow [N]$; $\Omega \leftarrow \emptyset$; draw a uniform permutation π of $[T]$.
while $|\mathcal{A}_r| > K + M$ **and** $(r-1)B < T$ **do**
  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. 6.
  $\tau_r \leftarrow (K+M)$-th largest LCB_i over $i \in \mathcal{A}_r$.
  $\mathcal{A}_{r+1} \leftarrow \{i \in \mathcal{A}_r : \mathrm{UCB}_i \geq \tau_r\}$; $r \leftarrow r + 1$.
end while
Rescore each survivor: $S_i \leftarrow \sum_{t=1}^{T} H_{i,t}$ via the fused MaxSim kernel.
return $\widehat{\mathcal{T}}_K \leftarrow \arg\operatorname{topK}_{i \in \mathcal{A}_r} S_i$.

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* $H \in \mathbb{R}^{N \times T}$ with entries $H_{i,t} \triangleq h(d_i, t)$ (the per-token MaxSim of Eq. 1), so each document's total late-interaction score is the row-sum $S_i \triangleq \sum_{t=1}^{T} H_{i,t} = S(d_i; Q)$ (Eq. 2). Our objective is to identify the set of indices $\mathcal{T}_K^\star$ corresponding to the K documents with the highest scores S_i (Eq. 3).

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 MaxSim cells revealed (each cell evaluation costs $O(L_d \cdot d)$ operations, so for a fixed corpus, coverage is proportional to MaxSim FLOPs; the small $K+M$ rescore is counted separately in the wall-clock results). At any time step of our algorithm the cost is:

$$\gamma(\Omega) \triangleq \frac{|\Omega|}{N \times T} \;=\; \frac{1}{NT} \sum_{i=1}^{N} |\mathcal{O}_i|. \tag{4}$$

4 Method: Col-Bandit

Overview. Figure 2 previews Col-Bandit: each round reveals a shared batch of B query tokens for the surviving documents only and tightens each survivor's interval $[\mathrm{LCB}_i, \mathrm{UCB}_i]$ (§4.1); a document is permanently eliminated once its upper bound falls below the $(K+M)$-th best lower bound; when only $K+M$ or fewer candidates remain, the reveal loop stops and they are rescored exactly. §4.2 gives the round-by-round procedure.

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 $LB_i^{\text{hard}} = \sum_{t \in \mathcal{O}_i} H_{i,t} + (T - n_i)\, a$ and $UB_i^{\text{hard}} = \sum_{t \in \mathcal{O}_i} H_{i,t} + (T - n_i)\, b$. 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 & Maillard, 2015),

$$r_i^{\text{eff}} \triangleq \alpha_{\text{ef}} \cdot T\widehat{\sigma}_i \sqrt{\frac{2\log(cNT/\delta)}{n_i}} \cdot \sqrt{\rho_{n_i}}, \tag{5}$$

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 B). The relaxation $\alpha_{\text{ef}} \in (0, 1]$ controls conservativeness:

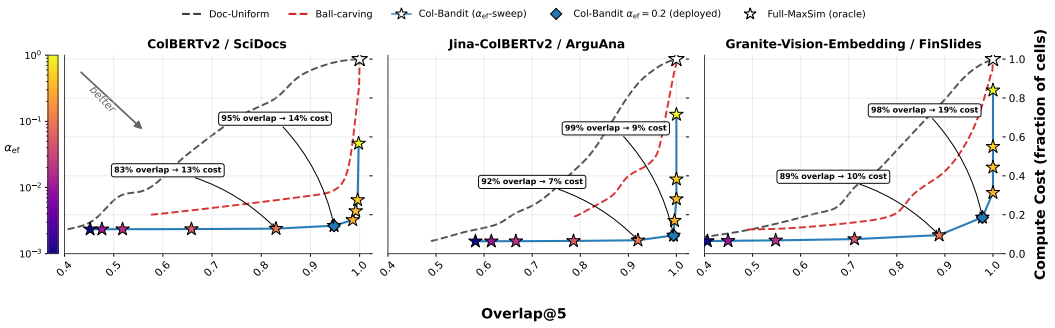

Figure 3: **Quality–coverage Pareto frontier (K=5).** Col-Bandit (sweeping α_{ef}) vs. Doc-Uniform and Ball-carving (MUVERA App. C.3) on ColBERTv2/SciDocs, Jina-ColBERTv2/ArguAna, and GVE/FinSlides. Col-Bandit dominates both baselines in every panel, reaching high fidelity at $\leq 20\%$ of the cell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

α_{ef}=1 removes the relaxation entirely (the unshrunk radius of Eq. 5), while α_{ef}<1 tightens it. The hybrid decision interval is

$$\begin{aligned} \text{LCB}_i &= \max(LB_i^{\text{hard}},\, \widehat{S}_i - r_i^{\text{eff}}), \\ \text{UCB}_i &= \min(UB_i^{\text{hard}},\, \widehat{S}_i + r_i^{\text{eff}}). \end{aligned} \tag{6}$$

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 *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 $(K{+}M)$-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 (annotated in Appendix F.1, traced on a real query in Appendix F.2). 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 E). 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 F). The Bernstein–Serfling pre-factor c in Eq. 5 (implemented at its exact value c=10) and the $n_i \leq 1$ case are detailed in Appendix B.

Conservative corner and deployed relaxation. Under uniform-without-replacement reveals, α_{ef}=1 admits a conditional δ-PAC guarantee (under the confidence radius of Eq. 5; Remark 2): $\mathbb{P}(\widehat{\mathcal{T}}_K{=}\mathcal{T}_K^\star) \geq 1{-}\delta$ (Theorem 1). The argument union-bounds the per-row Bernstein–Serfling intervals over all N documents and T reveal counts (the $\log(cNT/\delta)$ term in Eq. 5), making every $[\text{LCB}_i, \text{UCB}_i]$ valid *simultaneously*, hence also at the data-dependent stopping time, so on this $1{-}\delta$ event no true top-K member is ever eliminated (proof in Appendix E). For α_{ef}<1 the radii shrink and elimination is more aggressive, trading the conditional guarantee 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).

Table 1: **Universal Efficiency Analysis.** Mean coverage % (std across corpora) to recover **95% (near-lossless)** of Full-MaxSim's own Overlap@K / nDCG@K per dataset at $K \in \{5, 50\}$; lower is better. Averages match the per-dataset breakdowns in Appendix D.2 (90% threshold there too); NQ is not encoded for Jina-ColBERTv2 and excluded from its averages.

Method	Overlap@5	Overlap@50	nDCG@5	nDCG@50
ColBERTv2 (BEIR)				
Doc-Uniform	98% (3.8)	100% (0.0)	55% (20.6)	43% (22.0)
Ball-carving	42% (14.9)	41% (15.9)	28% (9.9)	24% (4.9)
Col-Bandit (Ours)	**14% (1.7)**	**18% (4.8)**	**13% (0.7)**	**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)**

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 an aggregate MM entry (the four REAL-MM-RAG corpora benchmarked separately and averaged, combined $N{=}8.6$ K) for the CPU wall-clock benchmarks (CPU-S and CPU-M1). Per-corpus details in Appendix C.2 (Table 3). N is the full corpus rerank size (e.g., $N{=}2.68$M 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 3.3 (IBM Research Team, 2025) 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 F.3).
Baselines. Full-MaxSim (exhaustive oracle), Doc-Uniform (non-adaptive random-cell reveal at coverage γ, Appendix C.3), Ball carving (Dhulipala et al., 2024) (Stage-2 query-time pruning peer, Appendix C.3), and four static token-selection orderings (uniform, query-token norm, IDF, and uniCOIL (Lin & Ma, 2021); §5.3). All methods are built on the same NUMKONG kernel and differ in what they reveal or compress per query (Appendix C.3).
Metrics. Our metric measures how Col-Bandit reproduces Full-MaxSim's top-K set:

$$\text{Overlap@}K \triangleq \frac{\left|\mathcal{T}_K^\star \cap \hat{\mathcal{T}}_K\right|}{K}, \tag{7}$$

where $\mathcal{T}_K^\star$ is the exhaustive top-K and $\hat{\mathcal{T}}_K$ is Col-Bandit's estimate, with $\text{Overlap@}K = 1$ iff the two sets coincide.[2] We report $K \in \{5, 50, 100\}$, coverage γ (Eq. 4), and per-query wall-clock latency. Task-level metrics (nDCG, Recall, MRR) are in Appendix D.4–D.5.
CB knobs. Deployed: $\alpha_{\text{ef}}{=}0.2$, $M{=}5$ (rescore margin), $\delta{=}0.01$. Round size $B{=}4$ matches the SIMD register tile of the fused C kernel. Pareto sweeps span $\alpha_{\text{ef}} \in (0, 1]$ (Appendix F.4). The $\alpha_{\text{ef}}{=}1$ corner admits the guarantee of Theorem 1 (scoped in Remark 2).

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.

[2]Throughout, "exact" means bit-identical to this exhaustive reference, namely the deployed Full-MaxSim kernel run on all $N \times T$ cells; the reference kernel itself (whose per-cell argmax uses an int8-assisted path) is audited against an independent pure-fp32 MaxSim in Appendix D.11.

Table 2: **Wall-clock CPU benchmarks on CPU-S** (AMD EPYC 7763, AVX2; **single thread**) at the deployed knob (α_{ef}=0.2, M=5, δ=0.01). **(a)** Per-query latency and speedup; each cell shows K=5 / K=100. Ov@K is overlap with Full-MaxSim's exhaustive top-K (Eq. 7), Cov the fraction of MaxSim cells revealed; the MM row averages the four REAL-MM-RAG corpora (combined N=8.6 K). Col-Bandit delivers a $\mathbf{7.0}\times$ / $\mathbf{5.3}\times$ mean speedup while preserving Ov@$K \geq 0.92$ on every corpus. **(b)** Retention of Full ColBERT's Top-1 metrics at matched coverage, nine-corpus mean (a protocol distinct from the wall-clock runs); MRR@1 equals nDCG@1 in every cell. Multi-thread breakdown: Table 17; per-query variance and bootstrap CIs: Appendix D.10; absolute scores: Appendices D.4, D.5.

(a) Latency and speedup

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	0.92 / 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×**

(b) Top-1 retention at matched coverage

Cov.	Metric	CB	Ball	Unif.
20%	R@1	**98.9%**	81.0%	55.9%
20%	nDCG@1	**98.7%**	82.1%	55.6%
40%	R@1	**99.1%**	90.8%	74.9%
40%	nDCG@1	**98.9%**	91.9%	74.6%

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 BEIR and REAL-MM-RAG. Col-Bandit dominates both non-adaptive baselines in every group average, and on Overlap@5 it wins every individual corpus; at larger K and on task metrics Ball-carving matches or slightly beats it on a few individual cells (per-corpus tables in Appendix D.2). Figure 3 confirms this across encoders and modalities: the α_{ef}-swept frontier lies strictly below both baselines in every panel, and the deployed α_{ef}=0.2 recovers ≈98% Overlap@5 at 19% cost on FinSlides. **Headline:** at the 95% near-lossless threshold, Col-Bandit reaches Overlap@5 at **14%/26%/22%** coverage on ColBERTv2/Jina/GVE respectively ($\mathbf{3.9}$–$\mathbf{7.1}\times$ savings), versus 98–100% for Doc-Uniform and 42–76% for Ball-carving, a gap of at least 2× on every corpus at K=5, narrowing to 1.6–1.9× on some corpora at K=50. 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 (≈$\mathbf{11}\times$) and 13% on ColBERTv2 ($\mathbf{7.7}\times$). Recall@5 and MRR@5 follow the same coverage-savings pattern (Col-Bandit reaches 95% retention at 7–24% coverage per corpus, with per-corpus tables in Appendix D.2). The looser 90% threshold is in Appendix D.1.

Wall-clock translation. Every method here shares the same fused C MaxSim kernel (NUMKONG, Appendix F.3) and differs mainly in which (d, t) cells it touches, and the bandit bookkeeping stays below 4% of runtime (Appendix D.6), so coverage savings translate to wall-clock via the heuristic Speedup ∼ 100/Cov%. Table 2 reports per-query latency on CPU-S (visualized as Figure 1; other α_{ef} and K values in Appendix D.3), and Table 16 (Appendix D.7) the CPU-M1 breakdown. 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 $\mathbf{7.0}\times$ mean single-thread speedup on CPU-S (K=5; 5.3× at K=100) and $\mathbf{12.9}\times$ on CPU-M1 (9.0× at K=100), where it is also $\mathbf{4.8}\times$ faster on average than *maxsim-cpu* (Clavié & Lee, 2025), the published Rust SIMD baseline, at comparable fidelity. End-task quality tracks the oracle: Col-Bandit retains ≥ 98.7% of Top-1 metrics at matched coverage (Table 2, right).

5.3 Sensitivity, Composability, and Calibration

Conservative corner vs. deployed knob. Table 19 (Appendix D.8) traces the speed–fidelity trade-off, holding kernel, quantisation, and memory layout fixed. Two controls delimit it. A *no-elimination* control (round structure and bookkeeping with elimination bypassed, rescore disabled) lands within 1.3–4.4% of Full-MaxSim's wall-clock, so the bandit bookkeeping is essentially free and all speedup is attributable to adaptive pruning. And the conservative corner α_{ef}=1 (Remark 2) is not merely theoretical: in a matched sweep (quality and latency from the same runs) it preserves Ov@5 ∈ [0.98, 1.00] on all nine corpora at 21–59% coverage on text (83–87% multimodal), a real 1.5–5.1× single-thread speedup (1.7× on SciDocs). Relaxing to the deployed knob trades the conditional guarantee for calibrated speed (6.3× on the same corpus). *Sensitivity in brief:* Col-Bandit is not a small-K specialization: latency scales sub-linearly in K; a 200× jump (K=5 → 1000)

costs only 1.3–4.5× (Table 23); the rescore margin $M{=}5$ lifts Ov@5 from 0.93–0.94 to ≥ 0.98 at +6–21% latency (Table 30); CPU-vs-GPU pairing and α_{ef} operating-point guidance: Appendix D.13.

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) ($d{=}128$) and its $d{=}64$ Matryoshka projection on four BEIR corpora (Table 1, middle block). 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**% ($\approx$**11**×). The same conclusion holds at $d{=}64$: 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 $d{=}128{\to}64$: Ov@5 retention ≥ 0.94 on every Jina-ColBERTv2 BEIR corpus and coverage essentially invariant to the projection; Appendix D.15) and *token-axis pooling* (Ward (Clavié et al., 2024): the pooling-induced loss dominates Col-Bandit's elimination cost in every measured cell; Appendix D.16). The axes stack with Col-Bandit's cell-skipping multiplicatively: *fewer dimensions* × *fewer tokens* × *fewer cells*.

Static token-selection baselines: adaptivity is the mechanism. Is per-document adaptivity necessary, or would a good *fixed* ordering of query tokens suffice? We score *all* documents on the top-m tokens of a fixed ordering (budget ratios 10–40% of query tokens), keep the top-M survivors, and exact-rescore (rescore cells counted in coverage), evaluating four orderings: uniform, query-token norm, IDF, and uniCOIL's learned query-term weights (Lin & Ma, 2021). Figure 8 (Appendix D.9) shows the resulting coverage–fidelity curves against Col-Bandit's α_{ef}-sweep on the same axis (both sides measured at $K{=}5$) on all five text corpora: every static ordering needs $\sim$40% coverage to reach Ov@5 of 0.74–0.98, while Col-Bandit reaches 0.92–0.97 at 13–15%; at matched coverage the best static ordering trails Col-Bandit by 8–46 points. The gap is structural: a static ordering spends the same budget on every document, whereas Col-Bandit concentrates its budget on candidates near the top-K boundary, information no query-side ordering can express.

Candidate-set reranking: composing with retrieval engines. Col-Bandit is a scoring layer, so the natural deployment question is composition with a first-stage retriever. Table 22 measures this directly: real PLAID (Santhanam et al., 2022a) indices (nbits $=2$) produce candidate sets of size K_{ann}, and Col-Bandit replaces the exhaustive rerank *on identical candidates* (single thread, $\alpha_{\text{ef}}{=}0.3$). Fidelity to the same-candidate exhaustive top-5 is 0.94–0.96 in every cell, and the speedup grows monotonically with candidate budget, from break-even at $K_{\text{ann}} \approx 100$ to 1.8–3.2× at $K_{\text{ann}}{=}1000$, approaching the full-corpus regime as $K_{\text{ann}} \to N$. The same protocol inside LEMUR's (Jääsaari et al., 2026) own pipeline leaves task quality effectively unchanged (Recall/nDCG within 0.004/0.003) while running end-to-end 1.5–2.0× faster at $k' \in \{1000, 2000\}$; a same-subset control confirms the gain is algorithmic, not kernel substitution (Appendix D.12). With *MUVERA-FDE* candidates, Col-Bandit preserves end-to-end nDCG@5 exactly (Ov@5 ≥ 0.999) at $R \in \{200, 500, 1000\}$, but these candidate sets sit below break-even, where Col-Bandit's round overhead makes the rerank itself slower (0.4 → 7.3 ms on SciDocs $R{=}200$), so composition there buys compatibility, not speed.

6 Conclusion

We presented Col-Bandit, an adaptive query-time framework that reveals only the MaxSim cells needed to resolve the top-K. The finding underneath is that late-interaction scoring is massively redundant: at the deployed setting, 13–22% of the MaxSim matrix suffices to preserve Ov@5 ≥ 0.90 on all nine corpora. Batched elimination under finite-population confidence bounds, realized in a fused SIMD kernel, turns this redundancy into wall-clock gains: up to $\sim$ **8**× fewer MaxSim FLOPs, for mean single-thread speedups of **7.0**× (server CPU) and **12.9**× (Apple M1 Max) at Ov@5 ≥ 0.90 on every corpus tested. One knob, α_{ef} (Eq. 5), controls the quality–compute trade-off (conditional δ-PAC guarantee at $\alpha_{\text{ef}}{=}1$; Theorem 1, Remark 2), and a single calibrated setting transfers across corpora, encoders, and modalities. Col-Bandit drops in with no retraining or index changes and composes with existing retrieval engines (limitations: Appendix A).

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.

Reproducibility statement

All experiments use public benchmarks (BEIR, REAL-MM-RAG) and public checkpoints. The algorithm is fully specified in §4 and Appendix pseudo-code; hyper-parameters and hardware are listed with each experiment. An anonymized repository containing the implementation and evaluation harness is provided for review at `https://anonymous.4open.science/r/ColBandit-3DBE`.

AI use statement

Large language models were used as assistants in this work: for drafting and editing manuscript text, for implementing and refactoring parts of the benchmark harness and analysis/plotting scripts (all LLM-assisted code was reviewed, tested, and validated against reference implementations by the authors), and for literature search. The core algorithm, its theoretical analysis, the experimental design, and all reported measurements are the authors' own; the authors take full responsibility for all content of this paper.

References

Jean-Yves Audibert and Sébastien Bubeck. Best arm identification in multi-armed bandits. In *COLT-23th Conference on learning theory-2010*, pp. 41–53, 2010.

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

Rémi Bardenet and Odalric-Ambrym Maillard. Concentration inequalities for sampling without replacement. *Bernoulli*, 21(3):1361–1385, 2015. doi: 10.3150/14-BEJ605. URL `https://arxiv.org/abs/1309.4029`.

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

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

Benjamin Clavié and Sean Lee. maxsim-cpu: Maximising Maxsim Efficiency, 2025. URL `https://www.mixedbread.com/blog/maxsim-cpu`.

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

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

Laxman Dhulipala, Majid Hadian, Rajesh Jayaram, Jason Lee, and Vahab Mirrokni. MUVERA: Multi-vector retrieval via fixed dimensional encodings. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*, 2024. URL `https://arxiv.org/abs/2405.19504`.

Shuai Ding and Torsten Suel. 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*, pp. 993–1002, 2011.

Joshua Engels, Benjamin Coleman, Vihan Lakshman, and Anshumali Shrivastava. DESSERT: An efficient algorithm for vector set search with vector set queries. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023. URL https://openreview.net/forum?id=kXfrlWXLwH.

Eyal Even-Dar, Shie Mannor, and Yishay Mansour. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of Machine Learning Research*, 7:1079–1105, 2006.

Ronald Fagin, Amnon Lotem, and Moni Naor. Optimal aggregation algorithms for middleware. In *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pp. 102–113, 2001.

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

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

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

Lixuan Guo, Yifei Wang, Tiansheng Wen, Aosong Feng, Stefanie Jegelka, and Chenyu You. No more k-means: Single-stage sparse coding for efficient multi-vector retrieval. In *International Conference on Machine Learning*, 2026. arXiv:2605.30120.

IBM Research Team. Granite-vision-3.3-2b-embedding, 2025. URL https://huggingface.co/ibm-granite/granite-vision-3.3-2b-embedding.

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

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

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

Omar Khattab and Matei Zaharia. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pp. 39–48, 2020.

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

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

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

Jimmy Lin and Xueguang Ma. A few brief notes on DeepImpact, COIL, and a conceptual framework for information retrieval techniques. *arXiv preprint arXiv:2106.14807*, 2021.

Rui Liu, Tianyi Wu, and Barzan Mozafari. A bandit approach to maximum inner product search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019.

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

Silvio Martinico, Franco Maria Nardini, Cosimo Rulli, and Rossano Venturini. Efficient multivector retrieval with token-aware clustering and hierarchical indexing. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2026. arXiv:2604.28142.

Nomic Team. Nomic embed multimodal: Interleaved text, image, and screenshots for visual document retrieval, 2025. URL https://nomic.ai/blog/posts/nomic-embed-multimodal.

Roi Pony, Daniel Ezer, Adi Raz Goldfarb, Idan Friedman, Oshri Naparstek, and Udi Barzelay. FLASH-MAXSIM: IO-aware fused kernels for late-interaction retrieval. *arXiv preprint arXiv:2605.29517*, 2026.

Keshav Santhanam, Omar Khattab, Christopher Potts, and Matei Zaharia. PLAID: An efficient engine for late interaction retrieval. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, CIKM '22, pp. 1747–1756, New York, NY, USA, 2022a. Association for Computing Machinery. doi: 10.1145/3511808.3557325.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3715–3734, Seattle, United States, 2022b. Association for Computational Linguistics.

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

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

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

Mo Tiwari, Ryan Kang, Je-Yong Lee, Donghyun Lee, Chris Piech, Sebastian Thrun, Ilan Shomorony, and Martin Jinye Zhang. Faster maximum inner product search in high dimensions. *arXiv preprint arXiv:2212.07551*, 2022.

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

Ash Vardanian. NumKong: 2000 Mixed Precision Kernels For All. URL https://github.com/ashvardanian/NumKong.

Henning Wachsmuth, Shahbaz Syed, and Benno Stein. 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)*, pp. 241–251, 2018.

Xiao Wang, Craig Macdonald, Nicola Tonellotto, and Iadh Ounis. 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*, pp. 2552–2561, 2023.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, et al. 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)*, pp. 2526–2547, 2025.

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

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

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

Jin Peng Zhou, Christian K. Belardi, Ruihan Wu, Travis Zhang, Carla P. Gomes, Wen Sun, and Kilian Q. Weinberger. On speeding up language model evaluation. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=3cvwO5DBZn.

Yuxuan Zong and Benjamin Piwowarski. Towards lossless token pruning in late-interaction retrieval models. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2025. arXiv:2504.12778.

A 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) concerns the $\alpha_{\text{ef}}{=}1$ corner with the implemented radius, which omits the additive $O(1/n)$ range term of the exact Bernstein–Serfling inequality (Remark 2). The deployed $\alpha_{\text{ef}}{=}0.2$ further shrinks the radius and makes elimination decisions on int8-quantised estimates, so it carries no formal certificate, trading it for the cost–fidelity operating point we report.

B 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 $\mathcal{O}_i$:

$$\widehat{\sigma}_i^2 = \frac{1}{n_i - 1} \sum_{t \in \mathcal{O}_i} \left(H_{i,t} - \widehat{\mu}_i\right)^2 . \tag{8}$$

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. (5) accounts for sampling without replacement from a finite set of size T. It is defined piecewise as:

$$\rho_n \triangleq \begin{cases} 1 - \dfrac{n-1}{T}, & n \leq T/2, \\ \left(1 - \dfrac{n}{T}\right)\left(1 + \dfrac{1}{n}\right), & n > T/2. \end{cases} \tag{9}$$

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 Constants and the $O(1/n)$ Range Term. The pre-factor inside the logarithm of Eq. (5) is instantiated at its exact value in the implementation: $c = 10$, the two-sided Bernstein–Serfling probability-mass constant of Bardenet & Maillard (2015, Theorem 4.3) (5 per side, doubled for the two-sided interval), combined with the full (document $\times$ reveal-count) union, so the implemented log term is $\log(10NT/\delta)$. The exact inequality additionally carries an additive $O(1/n)$ range term, which the implementation omits (Remark 2 scopes the guarantee accordingly). Moreover, the operative quantity is the *hybrid* interval $[\mathrm{LCB}_i, \mathrm{UCB}_i]$ (Eq. 6), 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.

C EXPERIMENTAL SETUP DETAILS

C.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. 2). The candidate set is the entire corpus throughout the main results (e.g. $N{=}2.68$M 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 the δ-PAC guarantee of Theorem 1 at $\alpha_{\text{ef}}{=}1$ (scoped in Remark 2).

Table 3: Evaluation dataset statistics. T_q: query token count; L_d: document token count (mean for text, fixed for vision).

Dataset	Corpus N	Queries	T_q	L_d	Encoder	Modality
BEIR (language)						
ArguAna	8,674	1.4K	32 / 64	192	ColBERTv2, Jina-ColBERT-v2	Text
SciDocs	25,657	1K	32 / 64	187	ColBERTv2, Jina-ColBERT-v2	Text
HotpotQA-500K	500,000	1K	32 / 64	68	ColBERTv2, Jina-ColBERT-v2	Text
Quora	522,931	5K	32 / 64	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	67.7	729	Granite Vision Embedding 3.3	Image+Text
FinReport	2,687	853	58.2	729	Granite Vision Embedding 3.3	Image+Text
TechSlides	1,963	1.4K	52.5	729	Granite Vision Embedding 3.3	Image+Text
TechReport	1,674	1.3K	61.4	729	Granite Vision Embedding 3.3	Image+Text

C.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 $d = 128$; ColBERTv2 uses a fixed query token length of $T{=}32$ and Jina-ColBERT-v2 of $T{=}64$. In addition, we evaluate Col-Bandit on a visual document retrieval task using the **REAL-MM-RAG** (Wasserman et al., 2025) benchmark, which includes four subsets: FinReport, FinSlides, TechReport, and TechSlides. In this setting, we employ the **Granite Vision Embedding 3.3**[5] (IBM Research Team, 2025) model, a vision-language embedding model that produces $d = 128$-dimensional token embeddings, with variable-length query representations and 729 document tokens per image. Table 3 summarizes the key statistics of all evaluation datasets. On the two sub-sampled candidate pools (HotpotQA-500K and Quora-Full), the evaluation query set is restricted to queries with at least one gold-relevant document inside the prepared subset (*qrel filter*): on a sub-sampled corpus, queries whose relevant documents all fall outside the subset make the exhaustive reference itself noisy. The filter is applied identically to Col-Bandit and to every baseline, so all comparisons remain matched.

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 3.3 (IBM Research Team, 2025)) 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 F.3).

C.3 Compared Methods

All compared methods operate, like Col-Bandit, on the Stage-2 candidate set (Appendix C.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

[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

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 $\arg\mathrm{topK}_{i \in [N]} \widetilde{S}_i$

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
  $\mathcal{C}_k \leftarrow \{i \in \mathcal{R} : \langle \hat{q}_p, \hat{q}_i \rangle \ge \tau\}$
  $c_k \leftarrow \sum_{i \in \mathcal{C}_k} q_i$
  $\mathcal{R} \leftarrow \mathcal{R} \setminus \mathcal{C}_k$
end while
// Step 2: rescore with one MaxSim term per cluster
for each $d \in \mathcal{D}$ **do**
  $\mathrm{Score}(d) \leftarrow \sum_{i=1}^{k} \max_{j \le L_d} \langle c_i, e_j^{(d)} \rangle$
end for
return $\arg\mathrm{topK}_{d \in \mathcal{D}} \mathrm{Score}(d)$

D Extended Experimental Results

D.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.

D.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 wins nearly every group average and the large majority of individual (corpus,

Table 4: **Universal Efficiency Analysis.** Mean coverage % (std across corpora) to recover **90% (high-fidelity)** of Full-MaxSim's own Overlap@K / nDCG@K per dataset at $K \in \{5, 50\}$; lower is better. Averages match the per-dataset breakdowns in Appendix D.2 (95% threshold there too); NQ is not encoded for Jina-ColBERTv2 and excluded from its averages.

Method	Overlap@5	Overlap@50	nDCG@5	nDCG@50
ColBERTv2 (BEIR)				
Doc-Uniform	91% (8.4)	92% (7.0)	37% (12.7)	28% (12.0)
Ball-carving	33% (10.7)	31% (8.9)	22% (5.5)	19% (5.1)
Col-Bandit (Ours)	**13% (0.9)**	**15% (2.4)**	**13% (0.1)**	**13% (0.5)**
Jina-ColBERTv2 (BEIR)				
Doc-Uniform	93% (7.8)	93% (7.8)	52% (26.6)	45% (24.4)
Ball-carving	37% (9.7)	34% (6.2)	16% (3.8)	16% (3.8)
Col-Bandit (Ours)	**17% (11.2)**	**28% (14.6)**	**7% (1.2)**	**9% (2.1)**
Granite-Vision-Embedding (REAL-MM-RAG)				
Doc-Uniform	83% (4.3)	83% (4.3)	34% (11.7)	27% (8.8)
Ball-carving	46% (2.1)	39% (5.3)	19% (5.3)	19% (5.3)
Col-Bandit (Ours)	**16% (4.8)**	**26% (5.4)**	**10% (1.5)**	**12% (1.3)**

Table 5: **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**). NQ is *n/a* for Jina-ColBERT-V2 (corpus not encoded at d=128) and excluded from its Average.

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)	**28%**	**39%**	**8%**	**11%**	*n/a*		**34%**	**40%**	**7%**	**9%**	**20% (11.8)**	**25% (14.7)**	**5.11×**	**4.04×**
$K = 50$														
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×**

K, metric) cells, with the exact magnitude varying with document length and query difficulty. On a handful of cells (mostly at $K \in \{10, 50\}$ and on task metrics, e.g., ColBERTv2/NQ, Jina/SciDocs at K=10), Ball-carving matches or slightly beats it, and Doc-Uniform edges ahead on one REAL-MM-RAG Recall@50 group average; the tables report these cases without aggregation.

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).

Table 6: **Per-dataset Efficiency Analysis: Overlap@K (REAL-MM-RAG).** 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**). Results are on the Granite-Vision-Embedding (GVE) multimodal benchmarks.

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×**

Table 7: **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 (threshold relative to Full-MaxSim's own score on that 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**). NQ is *n/a* for Jina-ColBERT-V2 (corpus not encoded at d=128) and excluded from its Average.

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%	22%	22%	22%	50%	50%	81%	50%	50%	37% (12.7)	55% (20.6)	2.71×	1.82×
Ball-carving	20%	26%	16%	19%	17%	17%	30%	34%	26%	44%	22% (5.5)	28% (9.9)	4.59×	3.56×
Col-Bandit (Ours)	**13%**	**14%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13% (0.1)**	**13% (0.7)**	**7.91×**	**7.66×**
Jina-ColBERT-V2														
Doc-Uniform	91%	100%	16%	16%	*n/a*		50%	70%	50%	50%	52% (26.6)	59% (30.7)	1.94×	1.70×
Ball-carving	15%	18%	10%	10%	*n/a*		20%	26%	19%	24%	16% (3.8)	20% (6.3)	6.30×	5.11×
Col-Bandit (Ours)	**10%**	**15%**	**6%**	**7%**	*n/a*		**7%**	**7%**	**7%**	**7%**	**7% (1.2)**	**9% (3.5)**	**13.48×**	**10.95×**
$K = 10$														
ColBERTv2														
Doc-Uniform	22%	41%	16%	22%	22%	50%	50%	81%	41%	72%	30% (13.1)	53% (21.4)	3.33×	1.88×
Ball-carving	20%	26%	16%	19%	12%	17%	23%	30%	26%	36%	19% (5.1)	26% (7.0)	5.22×	3.88×
Col-Bandit (Ours)	**13%**	**15%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13% (0.2)**	**13% (0.8)**	**7.87×**	**7.52×**
Jina-ColBERT-V2														
Doc-Uniform	70%	91%	16%	16%	*n/a*		50%	70%	41%	70%	44% (19.7)	62% (27.9)	2.27×	1.62×
Ball-carving	15%	18%	10%	10%	*n/a*		20%	26%	19%	30%	16% (3.8)	21% (7.8)	6.30×	4.74×
Col-Bandit (Ours)	**9%**	**28%**	**6%**	**7%**	*n/a*		**7%**	**8%**	**7%**	**7%**	**7% (0.9)**	**13% (9.0)**	**13.46×**	**7.91×**
$K = 50$														
ColBERTv2														
Doc-Uniform	22%	50%	16%	22%	22%	22%	50%	81%	31%	41%	28% (12.0)	43% (22.0)	3.56×	2.32×
Ball-carving	20%	26%	16%	19%	12%	17%	23%	30%	26%	26%	19% (5.1)	24% (4.9)	5.22×	4.21×
Col-Bandit (Ours)	**14%**	**14%**	**13%**	**13%**	**13%**	**13%**	**13%**	**14%**	**13%**	**14%**	**13% (0.5)**	**13% (0.6)**	**7.72×**	**7.54×**
Jina-ColBERT-V2														
Doc-Uniform	81%	91%	16%	16%	*n/a*		50%	50%	31%	70%	45% (24.4)	57% (27.7)	2.25×	1.77×
Ball-carving	15%	23%	10%	10%	*n/a*		20%	20%	19%	24%	16% (3.8)	19% (5.6)	6.30×	5.17×
Col-Bandit (Ours)	**13%**	**27%**	**7%**	**9%**	*n/a*		**10%**	**12%**	**8%**	**9%**	**9% (2.1)**	**14% (7.5)**	**10.88×**	**7.06×**

D.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

Table 8: **Per-dataset Efficiency Analysis: nDCG@K (REAL-MM-RAG).** We report the coverage budget required to achieve **90%** (white) and **95%** (gray) of Full-MaxSim's **NDCG@K** per corpus (threshold relative to Full-MaxSim's own score on that 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**). Results are on the Granite-Vision-Embedding (GVE) multimodal benchmarks.

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 9: **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 (threshold relative to Full-MaxSim's own score on that 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**). NQ is *n/a* for Jina-ColBERT-V2 (corpus not encoded at d=128) and excluded from its Average.

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%	12%	22%	22%	22%	72%	91%	50%	50%	39% (21.0)	47% (25.2)	2.54×	2.13×
Ball-carving	20%	26%	16%	19%	17%	17%	30%	34%	36%	44%	24% (7.9)	28% (9.9)	4.21×	3.56×
Col-Bandit (Ours)	**14%**	**14%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13% (0.7)**	**13% (0.7)**	**7.72×**	**7.66×**
Jina-ColBERT-V2														
Doc-Uniform	100%	100%	16%	16%	*n/a*		50%	50%	31%	50%	49% (31.7)	54% (30.1)	2.03×	1.86×
Ball-carving	18%	23%	10%	10%	*n/a*		20%	20%	19%	24%	17% (3.8)	19% (5.6)	6.03×	5.17×
Col-Bandit (Ours)	**15%**	**24%**	**6%**	**7%**	*n/a*		**7%**	**7%**	**7%**	**7%**	**9% (3.6)**	**11% (7.3)**	**11.16×**	**8.75×**
$K = 10$														
ColBERTv2														
Doc-Uniform	22%	25%	12%	12%	16%	22%	72%	81%	25%	72%	29% (21.7)	42% (28.3)	3.40×	2.35×
Ball-carving	26%	29%	16%	16%	12%	12%	23%	30%	36%	36%	22% (8.5)	25% (9.3)	4.46×	4.08×
Col-Bandit (Ours)	**15%**	**15%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13% (0.8)**	**13% (0.8)**	**7.57×**	**7.52×**
Jina-ColBERT-V2														
Doc-Uniform	50%	91%	6%	16%	*n/a*		31%	50%	20%	50%	27% (16.0)	52% (26.6)	3.71×	1.94×
Ball-carving	15%	18%	10%	10%	*n/a*		20%	20%	19%	19%	16% (3.8)	17% (3.8)	6.30×	6.03×
Col-Bandit (Ours)	**17%**	**28%**	**6%**	**7%**	*n/a*		**8%**	**8%**	**7%**	**7%**	**10% (4.2)**	**13% (9.0)**	**10.37×**	**7.91×**
$K = 50$														
ColBERTv2														
Doc-Uniform	16%	41%	12%	12%	12%	16%	31%	72%	25%	25%	19% (7.5)	33% (21.7)	5.16×	3.02×
Ball-carving	20%	26%	10%	16%	12%	12%	23%	23%	26%	26%	18% (6.1)	20% (5.9)	5.52×	4.90×
Col-Bandit (Ours)	**14%**	**17%**	**13%**	**13%**	**13%**	**13%**	**13%**	**14%**	**14%**	**14%**	**13% (0.6)**	**14% (1.6)**	**7.67×**	**7.15×**
Jina-ColBERT-V2														
Doc-Uniform	70%	91%	6%	16%	*n/a*		25%	31%	20%	31%	30% (24.0)	42% (28.7)	3.28×	2.37×
Ball-carving	23%	23%	8%	8%	*n/a*		15%	15%	19%	19%	16% (5.7)	16% (5.7)	6.15×	6.15×
Col-Bandit (Ours)	**16%**	**27%**	**7%**	**9%**	*n/a*		**10%**	**12%**	**9%**	**10%**	**11% (3.5)**	**14% (7.3)**	**9.51×**	**6.93×**

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.

Table 10: **Per-dataset Efficiency Analysis: Recall@K (REAL-MM-RAG).** We report the coverage budget required to achieve **90%** (white) and **95%** (gray) of Full-MaxSim's **Recall@K** per corpus (threshold relative to Full-MaxSim's own score on that 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**). Results are on the Granite-Vision-Embedding (GVE) multimodal benchmarks.

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 11: **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 (threshold relative to Full-MaxSim's own score on that 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**). NQ is *n/a* for Jina-ColBERT-V2 (corpus not encoded at d=128) and excluded from its Average.

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%	22%	22%	22%	25%	50%	50%	41%	50%	31% (11.9)	44% (18.4)	3.20×	2.29×
Ball-carving	20%	20%	16%	19%	17%	17%	23%	30%	26%	36%	20% (3.8)	24% (7.3)	4.94×	4.08×
Col-Bandit (Ours)	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13% (0.1)**	**13% (0.2)**	**7.95×**	**7.87×**
Jina-ColBERT-V2														
Doc-Uniform	91%	100%	16%	16%	*n/a*		50%	70%	50%	50%	52% (26.6)	59% (30.7)	1.94×	1.70×
Ball-carving	15%	18%	10%	10%	*n/a*		20%	26%	19%	24%	16% (3.8)	20% (6.3)	6.30×	5.11×
Col-Bandit (Ours)	**10%**	**10%**	**6%**	**7%**	*n/a*		**7%**	**7%**	**7%**	**7%**	**7% (1.3)**	**8% (1.1)**	**13.53×**	**13.17×**
$K = 10$														
ColBERTv2														
Doc-Uniform	22%	72%	22%	22%	22%	50%	50%	50%	41%	50%	31% (11.9)	49% (15.9)	3.20×	2.05×
Ball-carving	20%	20%	16%	19%	12%	17%	23%	30%	26%	36%	19% (5.1)	24% (7.3)	5.22×	4.08×
Col-Bandit (Ours)	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13% (0.1)**	**13% (0.3)**	**7.93×**	**7.82×**
Jina-ColBERT-V2														
Doc-Uniform	81%	100%	16%	16%	*n/a*		50%	70%	50%	70%	49% (23.2)	64% (30.5)	2.03×	1.56×
Ball-carving	15%	18%	10%	10%	*n/a*		20%	20%	19%	24%	16% (3.8)	18% (5.2)	6.30×	5.56×
Col-Bandit (Ours)	**9%**	**10%**	**6%**	**7%**	*n/a*		**7%**	**7%**	**7%**	**7%**	**7% (0.9)**	**8% (1.4)**	**13.55×**	**12.50×**
$K = 50$														
ColBERTv2														
Doc-Uniform	22%	81%	22%	22%	22%	25%	50%	50%	41%	50%	31% (11.9)	46% (21.4)	3.20×	2.19×
Ball-carving	20%	20%	16%	19%	12%	17%	23%	30%	26%	26%	19% (5.1)	22% (4.9)	5.22×	4.44×
Col-Bandit (Ours)	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13%**	**13% (0.3)**	**13% (0.3)**	**7.87×**	**7.83×**
Jina-ColBERT-V2														
Doc-Uniform	81%	100%	16%	16%	*n/a*		50%	70%	50%	70%	49% (23.2)	64% (30.5)	2.03×	1.56×
Ball-carving	15%	18%	10%	10%	*n/a*		20%	20%	24%	24%	17% (5.3)	18% (5.2)	5.79×	5.56×
Col-Bandit (Ours)	**10%**	**13%**	**6%**	**9%**	*n/a*		**10%**	**10%**	**8%**	**9%**	**8% (1.5)**	**10% (1.5)**	**11.89×**	**9.96×**

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 (conservative

Table 12: **Per-dataset Efficiency Analysis: MRR@K (REAL-MM-RAG).** We report the coverage budget required to achieve **90%** (white) and **95%** (gray) of Full-MaxSim's **MRR@K** per corpus (threshold relative to Full-MaxSim's own score on that 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**). Results are on the Granite-Vision-Embedding (GVE) multimodal benchmarks.

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×**

corner α_{ef}=1; Remark 2). 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.

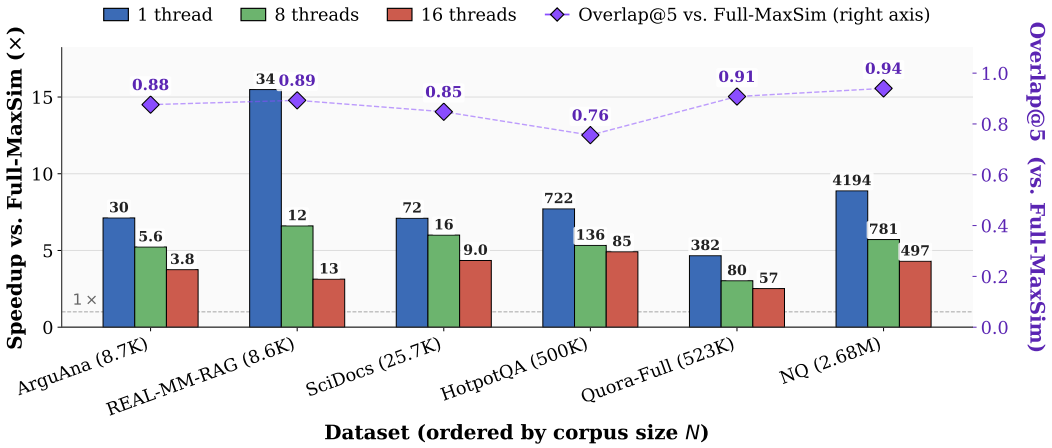

Figure 4: Wall-clock speedup at α_{ef}=0.1 (aggressive corner): higher speedup than the deployed knob but Ov@5 falls as low as 0.76 on the toughest corpus (full per-corpus ranges in Table 24).

K sweep at α_{ef}=0.2. Holding the deployed knob fixed, we grow 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 23): 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.

D.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 13 (referenced from §5.2) 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

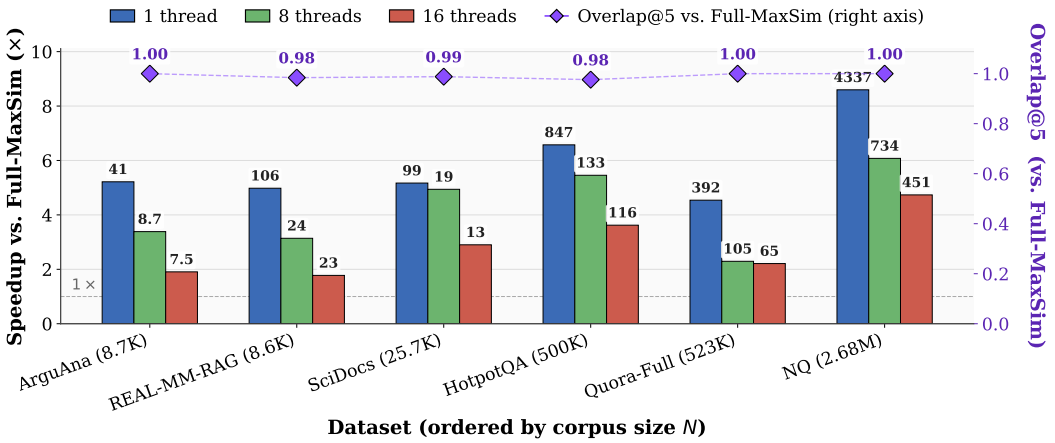

Figure 5: Wall-clock speedup at $\alpha_{\text{ef}}=0.3$ (conservative): a looser confidence radius than the deployed knob; Ov@5 ≥ 0.97 on every corpus at modestly reduced speedup.

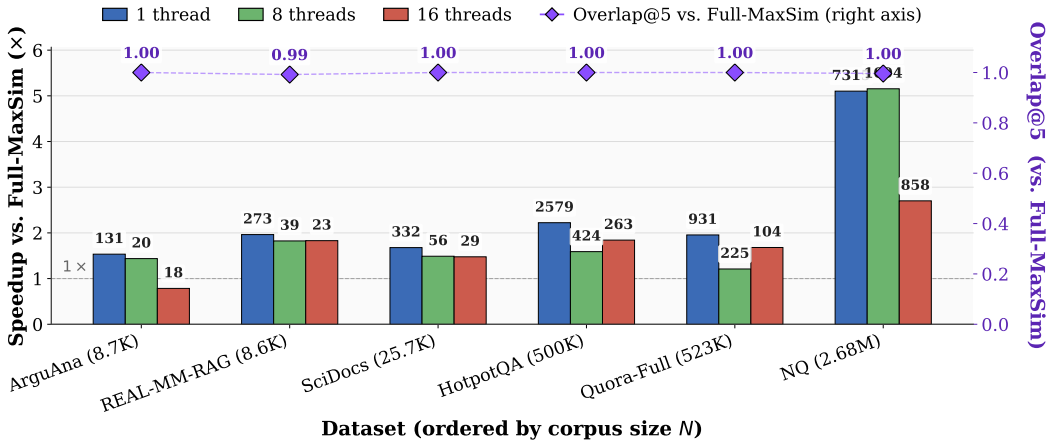

Figure 6: Wall-clock speedup at $\alpha_{\text{ef}}=1$ (the conservative corner of Theorem 1, Remark 2): Ov@5 $\geq$ 0.996 on every text corpus, at the cost of 21–59% coverage. Even at this conservative setting Col-Bandit delivers nontrivial speedups over Full-MaxSim.

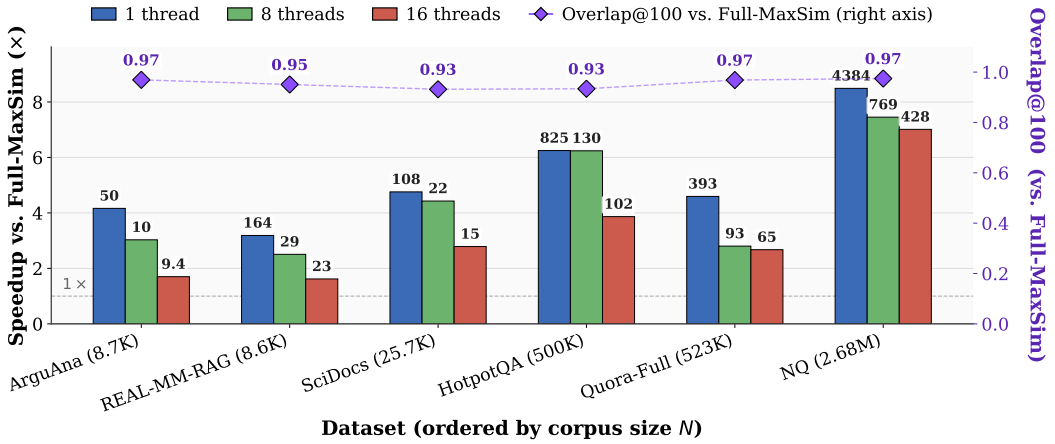

Figure 7: Wall-clock speedup at the deployed $\alpha_{\text{ef}}=0.2$, $K=100$ (recall-class setting): Col-Bandit retains substantial speedup at the larger top-K target with Ov@100 ≥ 0.93 on every corpus.

Table 13: 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 C.2). Full reranking at 100% coverage serves as the reference. Displayed values are rounded to two decimals; retention percentages quoted in the text are computed on the unrounded corpus-means.

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%

sampling for correctly identifying the single best document with high confidence. Per-corpus $K{=}5$ task metrics for BEIR are in Appendix D.2.

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

Table 14 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 76–99% for Ball-carving and Doc-Uniform; the gap widens as the coverage budget tightens, underscoring the value of variance-aware sampling for high-fidelity ranking.

D.6 Runtime Overhead Breakdown

Table 15 decomposes per-query Col-Bandit latency into three stages: the MaxSim micro-kernel calls during elimination rounds (kernel internals in Appendix F.3); 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; the shortfall is inside the kernel itself (bookkeeping is 2.5% of runtime, Table 15): on Quora's very short documents the surviving-cell pattern utilises the SIMD register tile poorly, so kernel time shrinks less than proportionally to coverage. In contrast, MM at 21% Cov yields $8.1\times$ at 1 t (the cell-skip pattern aligns favourably with the tile). Increasing α_{ef} from

Table 14: **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 76–99% 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 15: **Col-Bandit per-stage runtime breakdown on CPU-S at $K{=}5$, 1 thread.** Deployed knob ($\alpha_{\text{ef}}{=}0.2$, $M{=}5$). The kernel column is the MaxSim micro-kernel (kernel internals in Appendix F.3); 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%

the deployed 0.2 to the $\alpha_{\text{ef}}{=}1$ corner trades additional coverage for higher fidelity, as shown by the full Pareto frontier in Figure 3.

Table 16: **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 $\mathbf{12.9}\times$ / $\mathbf{9.0}\times$ ($K{=}5$ / $K{=}100$) mean single-thread speedup vs. Full-MaxSim and $\mathbf{4.8}\times$ / $\mathbf{3.5}\times$ vs. *m-cpu* (Clavié & Lee, 2025) (Rust SIMD baseline). Overlap@K can differ slightly from CPU-S because int8 dot products round differently across SIMD back-ends (Appendix F.3). The MM row averages the four REAL-MM-RAG corpora benchmarked separately (combined $N{=}8.6$ K). Threads parallelise across documents within a single query, so multi-thread figures remain genuine per-query latencies. 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×**

Table 17: **Wall-clock CPU benchmarks on CPU-S, multi-thread breakdown** (AMD EPYC 7763, AVX2; $\alpha_{\text{ef}}{=}0.2$, $M{=}5$, $\delta{=}0.01$). Per-query latency in ms (1t / 8t). Threads parallelise across documents *within* a single query, so multi-thread figures remain genuine per-query latencies. 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	0.92	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×**

D.7 Multi-Thread Wall-Clock Scaling

The main-paper wall-clock table (Table 2) reports single-thread latency for compactness; Table 16 gives the corresponding CPU-M1 (Apple M1 Max, NEON) cross-platform breakdown referenced from §5.2. 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 bookkeeping do not parallelise as cleanly as the dense kernel sweep, but Col-Bandit retains a substantial multi-thread speedup on every corpus.

D.8 Conservative Operating Point per Corpus

Table 20 expands the conservative ($\alpha_{\text{ef}}{=}1$) row of Table 19 per corpus. At $\alpha_{\text{ef}}{=}1$ with the Theorem 1 radius (Remark 2), Col-Bandit preserves Ov@5 $\in [0.98, 1.00]$ on every corpus. Coverage at termination is corpus-dependent: large text corpora terminate after revealing as little as 21–43% of the matrix (NQ, Quora, HotpotQA), while the small multimodal corpora, whose candidate pools are dominated by near-boundary documents, require 83–87%. Latency follows coverage: the conservative corner is a real speedup, not a formality (5.1× on NQ, 1.5–1.7× on the smaller text corpora).

D.9 Token-Selection Baselines on All Text Corpora

Figure 8 reports the coverage–fidelity comparison of §5.3 on all five text corpora. Note that the *uniform* ordering with top-M rescore is precisely a bounds-free reveal-then-rescore cascade (same reveals, same rescore, no confidence-bound elimination), so Col-Bandit's margin over it at matched coverage isolates the contribution of the statistical machinery itself, complementing the no-elimination wall-clock control of §5.3. The picture is uniform in budget terms: on the harder corpora every static ordering plateaus well below the 0.95 near-lossless line even at 40% coverage (HotpotQA: 0.74–0.77), and on the easier corpora the best static ordering reaches 0.94–0.98 only at 40% coverage, roughly 3× the budget Col-Bandit uses. Col-Bandit reaches 0.92–0.97 at 13–15% coverage everywhere, and at matched coverage the best static ordering trails it by 8–46 points. uniCOIL and IDF require

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 & Lee, 2025) is 1t only. Single-thread summary in Table 16.

		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: **Conservative (α_{ef}=1) vs. deployed operating points.** Ov@5 and coverage ranges span all nine corpora; wall-clock is SciDocs, single thread on CPU-S. Each row's quality and latency come from the same run: the conservative row is a matched α_{ef}=1 sweep (its own Full-MaxSim reference: 557 ms on SciDocs), the deployed row is the Table 2 run family. The no-elimination control runs the round structure (rescore disabled) with elimination bypassed, isolating the bookkeeping overhead. The conservative corner already yields a real speedup (1.5–5.1× across corpora); the deployed knob trades the theorem-scoped guarantee for calibrated speed.

Setting	**Guarantee**	**Ov@5**	**Coverage**	**SciDocs (ms)**	**Speedup**
Full-MaxSim	exact (oracle)	1.000	100%	513	1.0×
+ no-elimination control	—	—	100%	≈Full +1.3–4.4%	≈1×
CB (α_{ef}=1)	Thm. 1 radius (Remark 2)	0.98–1.00	21–59% text / 83–87% MM	332	1.7×
CB-Deployed (α_{ef}=0.2, M=5)	calibrated	$\geq$ 0.92	13–21%	81	6.3×

text-term statistics and are therefore undefined on the image-document REAL-MM-RAG corpora, which are omitted.

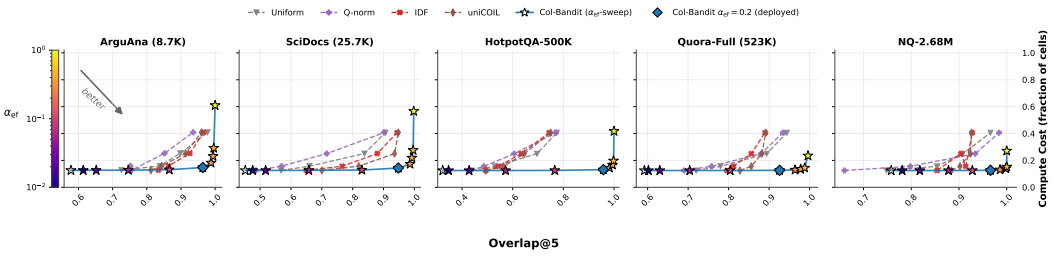

Figure 8: **Coverage–fidelity curves on all five text corpora** (both sides at K=5; axes and markers as in Figure 3: star colour encodes α_{ef}, diamond marks the deployed knob). Static orderings: budget ratios 10–40% of query tokens, top-M=25 survivors exact-rescored; Col-Bandit: α_{ef}-sweep; 100 queries per corpus.

D.10 Query-Level Variance and Seed Sensitivity

The main-text tables report seed-averaged means. Table 21 summarizes the distributional picture from fresh 100-query × 5-seed sweeps per corpus (50 × 3 on NQ-2.68M) at the deployed knob (the reveal order is the only stochastic component; Table 2 reported the frozen single-stream point, so small differences reflect query/seed pools at the same knob). Average behavior is consistent across seeds, and the tails are bounded: the worst single-seed corpus mean is 0.894, and fidelity misses typically replace a single document of the five, spread across queries rather than concentrated in a few failures.

D.11 Precision Path: int8 Elimination Audit

Both Full-MaxSim and Col-Bandit dispatch to the same i8-coarse → fp32-refine kernel, so quantisation and memory layout are held constant across every matched comparison in this paper. Within Col-Bandit, int8 arithmetic only *selects* the top-2 argmax candidates per cell; every $H[i,t]$ value entering the confidence bounds is recomputed in fp32, and the final $K+M$ rescore is bit-identical to Full-MaxSim. Three controls bound the int8 contribution. (i) Running the *entire* elimination loop in pure fp32 leaves Ov@5 unchanged on SciDocs and within 0.7 pp on HotpotQA-500K, at ∼2.5× slower elimination, so int8 is a speed optimisation, not a fidelity mechanism. (ii) At the conservative corner (α_{ef}=1), int8 and fp32 elimination agree to mean Ov@5 ∈ $[0.994, 1.000]$ on all corpora

Table 20: **Conservative operating point (α_{ef}=1) per corpus**, single thread on CPU-S. Ov@5 is fidelity to the exhaustive top-5; Cov is the fraction of MaxSim cells revealed at termination. All columns of a row come from the same matched run. Fidelity is near-exact everywhere while still skipping 41–79% of the matrix on text corpora; the multimodal corpora terminate at higher coverage.

Corpus	N	Ov@5	Cov	α_{ef}=1 (ms)	Full (ms)
ArguAna	8.7 K	1.000	57.7%	131	201
SciDocs	25.7 K	1.000	58.9%	332	557
Quora	522.9 K	1.000	27.7%	931	1,820
HotpotQA	500 K	1.000	42.7%	2,579	5,733
NQ	2.68 M	0.996	21.2%	7,315	37,324
FinSlides	2.3 K	0.992	83.0%	309	615
FinReport	2.7 K	1.000	87.4%	344	701
TechSlides	2.0 K	0.980	86.0%	223	419
TechReport	1.7 K	0.996	86.8%	215	405

Table 21: **Variance summary at the deployed knob** (100 queries × 5 reveal-order seeds per corpus, 50 × 3 on NQ-2.68M, K=5).

Statistic	Result
Latency p50 / p95	Text corpora: p50 within ∼3% of mean, p95/p50 ≤ 1.31; multimodal corpora carry heavier tails (p95/p50 2.0–2.1, p50 8–10% below mean; max 2.13, FinSlides)
Ov@5 query-level bootstrap 95% CIs	e.g. ArguAna 0.981 [0.972, 0.990]; HotpotQA-500K 0.922 [0.904, 0.942] (seeds averaged per query)
Seed sensitivity	Ov@5 std 0.5–3.4 pp across seeds (median 1.3 pp); worst single-seed corpus mean 0.894
Coverage p50 / p95 / p99	12.5–14.0 / 12.8–16.8 / 13.0–19.7% on large text corpora; MM wider (p99 ≤ 57%)
Exact top-5 rate	80–93% of (query, seed) runs return the exact top-5; misses typically replace one document of the five

measured, and where they differ the direction is not systematic, consistent with measurement noise rather than quantisation bias. (iii) Finally, we compare the exhaustive reference kernel itself against an independent pure-fp32 PyTorch MaxSim implementation over the full corpus (50–100 queries per corpus): top-5 agreement is 0.85–0.95 with argmax agreement 0.81–1.00, and per-document scores agree to a mean absolute drift of 0.12–0.49 (≈0.5–2% of the score scale), attributable to the i8 argmax-selection path quantified above. Where the two top-5 sets differ, the summed fp32-score gap between them is 0.02–0.34, i.e., the disagreements are confined to near-tied boundary documents rather than reflecting systematic mis-scoring. Completing the decomposition end-to-end (five corpora, 50 queries each, α_{ef}=0.2, K=5): the reference kernel agrees with pure-fp32 exhaustive MaxSim at Ov@5 0.82–0.96, deployed Col-Bandit agrees with the same fp32 reference at 0.81–0.93, and with the shared kernel at 0.94–0.98; elimination therefore adds only 0.4–5.6 points beyond the kernel's own drift, so the two error sources are separable and pruning is the smaller term. Because Full-MaxSim and Col-Bandit share this kernel, all Overlap@K comparisons in the paper measure the estimator against its own scorer's exhaustive answer and are unaffected by this reference-level drift.

D.12 LEMUR SAME-SUBSET KERNEL CONTROL

Table 22 (referenced from §5.3) reports the PLAID candidate-set replacement sweep.

The LEMUR composition result (§5.3) replaces LEMUR's exact-MaxSim rerank stage with Col-Bandit on identical top-k' candidate sets. To confirm the end-to-end gain is algorithmic rather than kernel substitution, we time LEMUR's optimized AVX-512 reranker against our exhaustive NUMKONG kernel on the same candidate subsets (50 queries, single thread): LEMUR's reranker is consistently *faster* than our exhaustive kernel (ratios 0.47–0.55 at $k' \in \{500, 1000, 2000\}$ on ArguAna and SciDocs). The 1.5–2.0× end-to-end speedup from inserting Col-Bandit is therefore achieved *against* a stronger-kernel baseline, and is attributable to adaptive cell revelation alone.

Table 22: **Candidate-set reranking: Col-Bandit composed with PLAID.** Rerank speedup (exhaustive rerank → Col-Bandit) over *identical* candidate sets of size K_{ann} produced by real PLAID indices (nbits =2), single thread, K=5. Ov@5 against the same-candidate exhaustive top-5 is 0.94–0.96 in every cell. Below $K_{\text{ann}} \approx 100$ the round overhead exceeds the savings and exhaustive rerank wins; the advantage then grows monotonically with candidate budget (SciDocs K_{ann}=1000: 20.8 → 9.4 ms), increasing toward the full-corpus regime of Table 2 (4.7–8.8× at full N).

K_{ann}	**ArguAna** (8.7**K**)	**SciDocs** (25.7**K**)	**Quora** (523**K**)
25	0.79×	0.81×	0.88×
100	1.32×	1.09×	1.04×
200	1.74×	1.35×	1.20×
500	2.50×	1.83×	1.50×
1000	**3.20**×	**2.22**×	1.82×
2000	—	—	**2.23**×

D.13 Operating-Point, K-Sensitivity, and Commodity-CPU Tables

These tables support claims made in §5.2 and §5.3 and are placed here for space.

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 23), 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 D.3 (Figure 7).

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 conservative corner of Theorem 1, Remark 2). Table 24 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 conservative corner. Notably, calibration *transfers*: the single deployed value α_{ef}=0.2 was chosen once and preserves Ov@5 ≥ 0.90 across all nine corpora, three encoders, and both modalities with no per-corpus tuning; per-corpus re-tuning is an available refinement, not a requirement. The headline-speedup chart at each $\alpha_{\text{ef}} \in \{0.1, 0.3, 1.0\}$ is in Appendix D.3 (Figures 4, 5, 6).

Commodity CPU vs. GPU at the deployed knob. Table 25 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 ≥ 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.

Kernels vs. skipping. Faster *exact* MaxSim execution (fused CPU kernels (Vardanian), optimized AVX-512 rerankers (Jääsaari et al., 2026), and IO-aware fused GPU MaxSim kernels (Pony et al., 2026)) reduces the cost of every scored cell, whereas Col-Bandit reduces *how many* cells are scored; the two axes are complementary and compose multiplicatively, as our NUMKONG-based realization demonstrates on CPU (§5.2). Our experimental focus is CPU reranking, where MaxSim dominates per-query latency and skipping pays most directly; on GPU, dense batching is efficient enough that fine-grained per-round elimination does not pay, and we leave a GPU-native realization (a fused MaxSim kernel with an in-kernel elimination point) to future work. In preliminary experiments on noise-expanded corpora, a prototype two-phase variant (a single elimination round using an earlier, simplified confidence radius) reached 3.8–5.2× over an unfused `torch.mm` Full-MaxSim baseline at

Table 23: 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 24: **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. Overlap lower endpoints are floored to two decimals so each range is a true bound. Smaller α_{ef} cuts more aggressively (lower coverage, lower fidelity); α_{ef}=1 is the conservative corner of Theorem 1.

α_{ef}	**Regime**	**Cov**	**Ov@5**
0.10	Aggressive	12.5–12.9%	0.75–0.91
0.20	**Deployed**	**12.6–14.4%**	**0.92–0.99**
0.30	Conservative	12.9–17.4%	0.97–1.00
1.00	Conservative	27.7–58.9%	1.00

≥ 0.90 top-K recall, though on the original un-expanded corpora a naive probe filter matched its quality at equal survivor counts.

D.14 Sensitivity to Query Length T and Choice of Round Size B

Query length T. The paper's two text encoders run at two query lengths (ColBERTv2 pads to T=32 and Jina-ColBERT-v2 to T=64; Table 3) on the same four BEIR corpora (SciDocs, Quora-Full, HotpotQA-500K, ArguAna). Because the encoders differ in more than T (training data, scoring distributions), this is *evidence from two encoders at two query lengths*, not a controlled sweep of T; we report the observed contrasts and mark causal readings explicitly as hypotheses. The mechanism does not degrade at the longer query length; the savings shift in *where* they appear. At the deployed knob (α_{ef}=0.2, M=5, K=5), mean coverage over the four shared corpora rises only mildly, from 13.7% under the T=32 encoder to 18.5% under the T=64 encoder, at essentially unchanged fidelity (mean Ov@5 0.949 vs. 0.945); and since the $N \times T$ matrix itself is twice as large at T=64 (Full-MaxSim wall-clock grows 1.7–2.2× on the same corpora), comparable relative savings are worth roughly twice the absolute compute. For *set identification* (Overlap@5 at the 95% threshold), the T=64 encoder is more corpus-dependent: the four-corpus mean rises from 14.5% to 25.8% coverage, with per-corpus spread widening from 13–17% to 8–49% (cheaper on ArguAna and Quora, dearer on SciDocs and HotpotQA; Table 5). One hypothesis, which this two-encoder design cannot isolate from encoder effects, is that longer queries spread score mass over more tokens, shrinking rank-boundary gaps relative to per-token increments on some corpora. For *task metrics* the trend reverses: the coverage to reach 95% of Full-MaxSim's nDCG@5 *falls* from 13.3% (the T=32 encoder) to 9.0% (the T=64 encoder) on the same four corpora (Table 7): under the longer-query encoder, fewer revealed tokens per document suffice where it matters for end-task quality. Finally, Col-Bandit does not rely on a fixed or padded T: GVE queries are variable-length ($T_q \approx$ 52–68; Table 3) and the REAL-MM-RAG block of Table 1 shows the same behavior (22% coverage for 95% Overlap@5, 15% for nDCG@5).

Round size B. B=4 is a design choice matched to the implementation, not a tuned hyperparameter: the fused kernel's inner loop processes a 4×4 register tile, a batch of four query tokens against four document-token positions per step (Appendix F.3), so one elimination round is one fused pass over the surviving rows. Two consequences of B are structural, following from the algorithm alone: the number of rounds is $\lceil T/B \rceil$, and elimination decisions occur only at multiples of B revealed tokens, so a surviving document can overshoot its earliest elimination point by up to $B-1$ tokens (at T=32, B=8 would halve the decision points from 8 to 4). The *performance* consequences of other values of B are unmeasured (we ran no B-sweep), so we do not claim B=4 is optimal, only that it matches the kernel's tile; the same value is used in every experiment in the paper.

Table 25: **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 26: 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. Coverages here come from an earlier sweep protocol and query pool and are not directly comparable to the deployed-sweep coverages of Appendix D.14; the within-table d=128 vs. d=64 comparison is matched.

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.

D.15 MATRYOSHKA DIMENSION REDUCTION

Jina-ColBERTv2's d=128 → d=64 Matryoshka projection halves the per-token storage with no impact on Col-Bandit's elimination dynamics. Table 26 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 of four corpora; on Quora-Full coverage actually *drops* 5.9 pp at d=64 with no quality loss). Ov@5 retention is ≥ 0.94 on every cell (≥ 0.95 on seven of eight); nDCG@5 retention is 100% on five of eight cells and $\geq 93.5\%$ everywhere. The dimension and cell-skipping axes are orthogonal: halving d does not change which documents Col-Bandit eliminates.

D.16 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 shrinks the document-token axis at encode time, making each MaxSim cell (a max over L_d/k instead of L_d document tokens) cheaper; Col-Bandit then eliminates documents at query time exactly as before. We benchmark the stack on CPU-S for SciDocs (text) and TechSlides (multimodal) at the deployed knob.

Table 27 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

Table 27: **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 in every cell ($\sim$10$\times$ in the medians).

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**

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$\times$ wall-clock speedup vs Ward-Full on the pooled corpus.

E 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. 5)**.** *For any $\delta \in (0,1)$ and α_{ef}=1, under uniform-without-replacement reveals within each row, and* provided *the radius of Eq. 5 is a valid per-row $(1-\delta_{i,n})$ confidence radius (Remark 2 discusses the omitted $O(1/n)$ range term), Algorithm 1 returns a set $\widehat{\mathcal{T}}_K$ satisfying*

$$\mathbb{P}\Big(\widehat{\mathcal{T}}_K \;=\; \mathcal{T}_K^\star\Big) \;\geq\; 1-\delta. \tag{10}$$

Remark 2 (Scope of the guarantee). Theorem 1 holds for the decision radius *as implemented* at α_{ef}=1: the exact two-sided Bernstein–Serfling pre-factor (10 inside the logarithm) and the exact piecewise ρ_n of Bardenet & Maillard (2015, Theorem 4.3) are retained, while the additive $O(1/n)$ range term of the exact inequality is omitted (Appendix B); the omission is an empirically calibrated choice and carries no confidence guarantee, so the validity of the implemented radius is the theorem's standing premise; restoring the omitted term recovers an unconditional guarantee at higher compute. As a sanity check, the α_{ef}=1 corner attains mean Overlap@5 ≥ 0.996 on every text corpus (1.00 on all but NQ; Appendix D.3). The theorem also concerns the *exact-precision* reveal model: the deployed CB-NK kernel makes elimination decisions on int8-quantised estimates (Appendix F.3), a further empirical relaxation for which we make no PAC claim. We therefore describe α_{ef}=1 as δ-PAC *under the implemented radius in the exact-precision model*, and treat the deployed α_{ef}=0.2 (with int8 elimination) as a calibrated relaxation (Limitations).

Why one shared permutation? The reveal schedule is a single permutation π of $[T]$, shared by every surviving document. We note upfront that sharing is an *implementation choice*, not a statistical necessity: Theorem 1's union bound already covers every (document, reveal-count) pair, so any schedule fixed *before* any entry of H is observed (independent per-document permutations

included) keeps each row a bona fide uniform without-replacement sample, and interval validity at the data-dependent stopping time follows just as it does here. What the analysis does require is that revealed cells are never chosen as a function of observed values; only observation-*dependent* token choice would break the sampling model. The shared schedule is preferred on practical grounds. *(i) Common reveal counts.* Under it, every active row has revealed the same n tokens, so the Bernstein–Serfling intervals entering each cross-document comparison of the elimination rule sit at a common reveal count n, the simplest instance the union bound covers. *(ii) Hardware.* Because all survivors reveal the same B tokens, each round is one contiguous SIMD pass: the $B{=}4$ query tokens stay in vector registers while document tokens stream through once (Appendix F.3). Per-document token choice would instead require gathers over an irregular access pattern, giving up the contiguous layout and batching that let the reveal kernel match Full-MaxSim's per-cell cost. *(iii) Where adaptivity lives.* Col-Bandit's adaptivity is in the per-document *stopping time*, not in per-document cell selection: an eliminated document simply stops paying for further tokens, which is what produces the coverage reductions of §5.2. Independent per-document (observation-independent) permutations would therefore be equally valid, but would forfeit the single-pass kernel and the common-n comparisons, and we would not expect a fidelity gain in exchange: token order affects only *when* a document's interval separates, not the exhaustive scores being estimated.

We now make the premise of Theorem 1 precise: we derive the time-uniform confidence statement in the uniform-sampling mode for the implemented radius of Eq. 5 at $\alpha_{\mathrm{ef}} = 1$. 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 $H_{i,t} \in [a, b]$, the empirical Bernstein–Serfling inequality of Bardenet & Maillard (2015, Theorem 4.3) yields a two-sided radius of this form up to an additive $O(1/n)$ range term (Appendix B). The *implemented* radius below keeps its pre-factor ($\log(10NT/\delta)$, allocated as $\delta_{i,n} = \delta/(NT)$ per document–reveal-count pair, two-sided) and the exact piecewise ρ_n, and omits that additive term; its validity as a $(1 - \delta_{i,n})$ confidence radius is the standing premise of Theorem 1:

$$\Pr\left(|S_i - T\widehat{\mu}_i| \;\le\; T\widehat{\sigma}_i\sqrt{\frac{2\log(10/\delta_{i,n})}{n}}\sqrt{\rho_n}\right) \;\ge\; 1 - \delta_{i,n} \qquad \text{(premise)}.$$

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,

$$S_i \in \left[T\widehat{\mu}_i \pm r_i^{\mathrm{th}}(n)\right], \quad r_i^{\mathrm{th}}(n) \triangleq T\widehat{\sigma}_i\sqrt{\frac{2\log(10NT/\delta)}{n}}\sqrt{\rho_n}.$$

In this uniform-within-row mode, choosing $\alpha_{\mathrm{ef}} = 1$ in Eq. 5 recovers exactly this form; the exact inequality additionally carries an additive $O(1/n)$ range term (Appendix B).

Proof sketch of Theorem 1. We give the high-level argument for the $\alpha_{\text{ef}}=1$ setting with uniform-without-replacement reveals, under the premise of Theorem 1.

(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 & 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 $\mathrm{LCB}_i \le S_i \le \mathrm{UCB}_i$. With $\alpha_{\text{ef}} = 1$ the radius in Eq. 5 is, by the theorem's premise, a valid per-row confidence radius (its relation to the exact Bardenet–Maillard form is discussed in Remark 2 and Appendix B), so $\mathbb{P}[A_r] \ge 1 - \delta_r$ 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 $[\mathrm{LCB}_i, \mathrm{UCB}_i]$ 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 $\mathbb{P}[\bigcap_r A_r] \ge 1 - \delta$.

(4) Elimination correctness. On $\bigcap_r A_r$, every surviving document obeys $\mathrm{LCB}_i \le S_i \le \mathrm{UCB}_i$. If at round r document d has $\mathrm{UCB}_d < \tau_r$, where τ_r is the $(K{+}M)$-th largest LCB_i over $\mathcal{A}_r$, then at least $K{+}M \ge K$ documents j satisfy $\mathrm{LCB}_j \ge \tau_r > \mathrm{UCB}_d$; on $\bigcap_r A_r$ each obeys $S_j \ge \mathrm{LCB}_j > \mathrm{UCB}_d \ge S_d$, so d is outranked by $\ge K$ documents, hence $d \notin \mathcal{T}_K^\star$ and removing it from $\mathcal{A}_{r+1}$ is safe. Following the Action/Successive Elimination argument of Even-Dar et al. (2006); Kalyanakrishnan et al. (2012); Audibert & 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 $|\mathcal{A}_r| \le K{+}M$ or $(r{-}1)B \ge T$. 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 $\arg\operatorname{topK}$ 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 (10) follows from $\mathbb{P}[\bigcap_r A_r] \ge 1 - \delta$ and the deterministic implication "$\bigcap_r A_r \Rightarrow \widehat{\mathcal{T}}_K = \mathcal{T}_K^\star$".

Empirical sanity check. At $\alpha_{\text{ef}}=1$ on every text corpus, measured mean Overlap@5 is ≥ 0.99 (Table 24, $\alpha_{\text{ef}}=1$ row), a per-query average consistent with the $\delta=0.01$ target, though not itself the full-set recovery probability of Eq. (10). Increasing coverage drives Overlap@5 toward 1.0 on every corpus, consistent with the radius collapsing as $n_i \to T$.

F Implementation Details and Parameter Selection

F.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. This observation-independent schedule keeps every row a uniform without-replacement sample, as the sampling model of Theorem 1 requires (sharing itself is an implementation choice; Appendix E).

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 $[\mathrm{LCB}_i, \mathrm{UCB}_i]$ via Eq. 6. The elimination threshold τ_r is the $(K{+}M)$-th largest lower bound over the active set (line 10); any document whose *upper* bound is below τ_r provably lies outside the top-$(K{+}M)$, hence

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

Require: Query Q (T tokens), candidate set $\mathcal{D} = \{d_1, \dots, d_N\}$, target K, margin M, batch B, calibration α_{ef}, confidence δ
Ensure: Estimated top-K set $\widehat{\mathcal{T}}_K \subseteq \mathcal{D}$
1: $\mathcal{A}_1 \leftarrow [N], \quad \Omega \leftarrow \emptyset, \quad r \leftarrow 1$ ▷ Active set; observed entries; round counter
2: $[\mathrm{LCB}_i, \mathrm{UCB}_i] \leftarrow [a\,T,\, b\,T]$ for all $i \in [N]$ ▷ Trivial initial interval
3: $\pi \leftarrow$ random permutation of $[T]$ (uniform, fixed seed; Appendix E)
4: **while** $|\mathcal{A}_r| > K + M$ **and** $(r{-}1)B < T$ **do**
5: $\quad \mathcal{B}_r \leftarrow \{\pi[(r{-}1)B{+}1], \dots, \pi[\min(rB, T)]\}$ ▷ Next B tokens of π
6: $\quad$ **for all** $i \in \mathcal{A}_r$ **in parallel** (one SIMD pass) **do**
7: $\quad\quad$ Reveal $\{H_{i,t} : t \in \mathcal{B}_r\}$; $\quad \Omega \leftarrow \Omega \cup (\{i\} \times \mathcal{B}_r)$
8: $\quad\quad$ Update $\widehat{S}_i$ and $[\mathrm{LCB}_i, \mathrm{UCB}_i]$ via Eq. 6 (radius $\propto \alpha_{\text{ef}}, \delta$)
9: $\quad$ **end for**
10: $\quad \tau_r \leftarrow (K{+}M)$-th largest LCB_i over $i \in \mathcal{A}_r$ ▷ Elimination threshold
11: $\quad \mathcal{A}_{r+1} \leftarrow \{i \in \mathcal{A}_r : \mathrm{UCB}_i \geq \tau_r\}$ ▷ Permanent elimination
12: $\quad r \leftarrow r + 1$
13: **end while**
14: **for all** $i \in \mathcal{A}_r$ **do** ▷ K-margin rescore on survivors
15: $\quad S_i \leftarrow \sum_{t=1}^{T} H_{i,t}$ using high-precision kernel
16: **end for**
17: **return** $\widehat{\mathcal{T}}_K \leftarrow \arg\mathrm{topK}_{i \in \mathcal{A}_r} S_i$

outside 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 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 F.3, Table 15).

F.2 A Worked Example on a Real Query

We trace Algorithm 1 on one real query at the deployed setting (α_{ef}=0.2, K=5, M=5, δ=0.01, B=4), using the exact-arithmetic reference implementation on ColBERTv2/SciDocs (N=25,657). The query is *"A Dual Prediction Network for Image Captioning"*; ColBERTv2 encodes it as T=32 query tokens: `[CLS]`, the query marker `[Q]`, nine WordPiece tokens (`a`, `dual`, `prediction`, `network`, `for`, `image`, `capt`, `##ion`, `##ing`), `[SEP]`, and twenty `[MASK]` expansion tokens. A single uniform permutation of the 32 positions is drawn once and shared by every document; each round reveals the next B=4 positions for the surviving documents only. Table 28 shows the run.

The loop has two exit conditions (Algorithm 1, line 2): it stops as soon as $|\mathcal{A}_r| \leq K{+}M$, *or* once all T tokens have been revealed. On easier queries the active set reaches $K{+}M$ after only a few rounds; the loop then exits early and the survivors proceed to the exact rescore with part of their cells still unrevealed. On this query the borderline candidates kept the loop running through all $T/B = 8$ rounds, reaching exactly $K{+}M$=10 survivors at the last decision point; by then every survivor was fully revealed, so no unrevealed cells remained (the deployed kernel's final $K{+}M$ rescore recomputes the survivors over all T tokens for bit-exact scores; its cost is counted separately from coverage). Had more than $K{+}M$ documents still been active after round 8, the algorithm would simply have rescored all of them exactly (they are fully revealed at that point) and returned the top-5; correctness never depends on reaching $K{+}M$. The total cost is 14.5% of the MaxSim cells, and the returned set is *identical* to the exhaustive top-5. The *New cells* column shows where the cost concentrates: round 1 (four tokens over all N documents, three of them `[MASK]` expansions) accounts for 86% of the total cost and already eliminates 89% of the corpus; the seven remaining rounds

Table 28: Round-by-round trace of Algorithm 1 on the query above ([M] = [MASK]). Each round reveals the next $B{=}4$ permuted token positions for the active documents only, so it costs $4\,|\mathcal{A}_r|$ new cells; a document is dropped once its UCB falls below the $(K{+}M)$-th best LCB. Cumulative coverage counts revealed cells as a fraction of the $N \times T = 821{,}024$ matrix cells.

Round	Tokens revealed	Active documents		New cells	Cum. cov.
1	[M] [M] capt [M]	25,657	$\rightarrow$ 2,705	102,628	12.5%
2	[M] [M] [CLS] ##ing	2,705	$\rightarrow$ 665	10,820	13.8%
3	[M] [M] for [M]	665	$\rightarrow$ 391	2,660	14.1%
4	[M] [M] [M] network	391	$\rightarrow$ 288	1,564	14.3%
5	[M] [M] [M] a	288	$\rightarrow$ 87	1,152	14.5%
6	[M] [M] [SEP] [Q]	87	$\rightarrow$ 37	348	14.5%
7	[M] [M] dual prediction	37	$\rightarrow$ 30	148	14.5%
8	[M] image [M] ##ion	30	$\rightarrow$ 10	120	14.5%

together add only 2 coverage points, spent on a few hundred borderline candidates: per-document adaptivity concentrated exactly where the decision is hard.

F.3 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 16 are apples-to-apples by construction: only the reveal schedule differs, not the kernel quality. Our NUMKONG-based Col-Bandit kernel (CB-NK) is available for review via an anonymized repository (`https://anonymous.4open.science/r/ColBandit-3DBE`) and will be released as open-source software upon publication.

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 16); (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 and Col-Bandit dispatch to the same i8-coarse $\rightarrow$ fp32-refine kernel, so quantisation and memory layout are held constant across all matched comparisons (Appendix D.11). Col-Bandit (CB-NK) uses the int8-quantised coarse pass only to select argmax candidates for the cheap LCB/UCB updates inside the elimination loop (each $H[i,t]$ entering the bounds is recomputed in fp32), 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.

Preprocessing and memory costs. The packed layout is produced by one offline, streamable linear pass over the corpus embeddings: 712 s for our largest corpus (NQ, 2.68M documents, $\sim$482M padded token slots; $\approx$1.5 s per million slots), and proportionally less for the others. On memory, the packed fp32 buffer *replaces* the original embedding storage, so the net addition is the int8 buffer plus per-document metadata, $\approx$27% of the fp32 footprint. In absolute terms the packed layout is often smaller than the padded fp32 tensors it replaces (0.29–0.79$\times$ across the large corpora; NQ: 148 GB packed vs. 247 GB padded). No index structure is built and no training is performed.

Multi-threading model. Parallelism is across documents *within* a single query via OpenMP: the elimination rounds and the final rescore partition the surviving documents across threads, and queries are benchmarked sequentially (one timed call per query). The 1-thread vs. 8-thread columns in

Table 17 (and the corresponding M1 figures) are therefore genuine per-query latencies at each thread count, matching the deployment regime of accelerating an individual query.

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 & 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 16; and GPU (NVIDIA A100 80 GB) for all GPU sweeps and the Pareto figure (Figure 3).

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 by the coarse argmax pass inside the elimination loop, and an aligned fp32 buffer used both to refine each revealed cell (the two int8-selected argmax candidates are recomputed in fp32 before entering the bounds; Appendix D.11) and 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 speed while all values entering bounds and scores use float arithmetic. Because the per-cell argmax *selection* is made in int8 (the refined values are fp32), quantization can change which document token attains a cell's max and hence *which* borderline documents survive to the rescore, while the survivor *scores* are always exact with respect to the deployed kernel. 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 & 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 16). 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

Table 29: **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 $\sim$11$\times$ faster than *maxsim-cpu*, and our NUMKONG Full-MaxSim is itself $\sim$2.3$\times$ 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 30: **K-margin rescore ablation** (α_{ef}=0.2, K=5, 1 thread, 50 queries per corpus). With M=0 only the top-K survivors are rescored, so near-boundary selection mistakes made on int8 estimates go uncorrected; M=5 restores Ov@5 $\geq$ 0.98 on both corpora at a moderate latency cost. Numbers are not directly comparable to the Ov@5 in Table 2, which uses a different query sample.

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

CPU-S Full timings of Table 2 (ArguAna 180 vs. 211 ms; SciDocs 509 vs. 513 ms), supporting it as an ISA-matched proxy. Table 29 reports the full breakdown.

F.4 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 monotone effect: increasing M from 0 to 5 adds 4–5 percentage points of Overlap@K at a modest runtime cost (+6–21% on the two corpora swept), since rescoring $K+M$ documents on all T tokens is a fixed overhead. We use M=5 by default and select α_{ef} based on a desired quality–coverage trade-off. Table 30 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. 5 (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. The key point is that α_{ef} is a single continuous knob with an interpretable conservative endpoint at α_{ef}=1 (Remark 2), so practitioners can dial in a desired quality–coverage operating point along a one-dimensional Pareto.

G 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.

G.1 Extended Related Work

This expands the condensed discussion in §2.2.

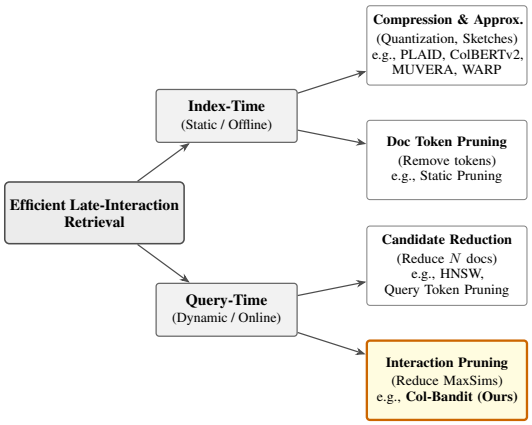

Figure 9: **Taxonomy of efficient late-interaction retrieval.** Methods are classified by *when* they prune (index-time vs. query-time) and *what* they prune. Col-Bandit is the first to dynamically prune the atomic interaction matrix H during query-time scoring.

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 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 discard non-informative tokens to shrink the document token index (Lassance et al., 2021) or the effective query length (T) at retrieval time (Tonellotto & Macdonald, 2021), 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 & 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 identify the Top-K set under statistical confidence bounds.

Exact MaxSim Kernels & Late-Interaction Serving Engines. A complementary line of work accelerates the *exact* MaxSim computation itself: Flash-MaxSim (Pony et al., 2026) fuses the similarity and row-max reduction into IO-aware GPU kernels that stream tiles through on-chip SRAM and never materialize the intermediate query-token $\times$ document-token similarity tensor, while maxsim-cpu (Clavié & Lee, 2025) plays the analogous role on CPU. These kernels make every cell cheaper but still evaluate *all* of them; Col-Bandit is orthogonal, deciding *which* cells to compute at all, and its per-round reveals can in principle execute through any such kernel. Recent serving engines such as TACHIOM (Martinico et al., 2026), a token-aware clustering and hierarchical indexing engine, and SSR (Guo et al., 2026), which replaces k-means centroid indexing with single-stage sparse coding, accelerate *candidate generation*, but score their retained candidates exhaustively; Col-Bandit stacks on either as the reranking layer. Trained token pruning, including recent near-lossless variants (Zong & Piwowarski, 2025), commits at index time to a reduced token set; Col-Bandit is zero-shot and query-adaptive: elimination decisions run on the quantized (int8) fast path and the $K+M$ survivors' final scores are bit-identical to the exhaustive reference.

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. 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 resolve 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), action elimination (Even-Dar et al., 2006), UCB-E (Audibert & 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 & 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., 2025), k-NN search (LeJeune et al., 2019).

H TABLE OF NOTATIONS

The notations used in the paper are described below.

Table 31: 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
Overlap@K	Headline fidelity metric: $\lvert\mathcal{T}_K^\star \cap \widehat{\mathcal{T}}_K\rvert / K$ (Eq. 7)
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. 5)
$\mathrm{LCB}_i, \mathrm{UCB}_i$	Hybrid lower/upper confidence bounds (Eq. 6)
$\mathcal{A}_r$	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: $(K+M)$-th largest LCB_i over $\mathcal{A}_r$
Algorithm Parameters	
$\alpha_{\mathrm{ef}} \in (0, 1]$	Calibration knob; α_{ef}=1 is the conservative corner (Theorem 1; Remark 2)
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. 5)