Storage-Equivalent Evaluation for Dense Retrieval: Placing Document Selection and Compression on a Common Per-Document Budget

Anonymous authors
Paper under double-blind review

Abstract

Dense-retrieval indexes are increasingly storage-bound. Two families of methods shrink them: *compression* spends fewer bits per document embedding, and *selection* (pruning) retains fewer documents. The field evaluates the two under incompatible protocols, compression over a fixed document set and selection at a fixed retention rate, and the gap is consequential: methods at equal retention can differ in per-document storage by up to 16×, so the two protocols can return opposite verdicts. We introduce *storage-equivalent evaluation*, which fixes a per-document bit budget B and lets each method retain $m = \lfloor BN/b \rfloor$ documents, placing pruning, quantization, and dimensionality reduction on one bits-per-document axis. The protocol returns a two-part answer. At a fixed compression code, intelligent selection reliably beats random: significantly, on all four core corpora, at both binary and PQ48×4. But choosing a cheaper code is the larger lever: across nine BEIR corpora and three encoders, a cheap product-quantization code beats every off-the-shelf selector measured at deployment budgets, and a closed-form dominance condition shows the disadvantage is structural, a selector must overcome the cheaper code's documents-per-bit advantage by document choice alone, before seeing the test queries, a bar learned selectors face no less. Among well-powered corpora selection beats compression on only one (BM25 selection on a financial corpus), but no measured corpus property (relevance density, intrinsic dimension, neighborhood similarity, lexical overlap, or the BM25-versus-dense retrieval gap) predicts where it helps.

1 Introduction

Dense retrieval underpins modern information access, from web search to retrieval-augmented generation (RAG) pipelines (Lewis et al., 2020). By encoding documents and queries as dense vectors and ranking by inner product, these systems are effective across domains (Karpukhin et al., 2020; Xiong et al., 2021), but effectiveness comes at a storage cost: a corpus of 100 million documents encoded as 768-dimensional FP32 vectors needs over 300 GB for embeddings alone. The constraint binds in practice (indexes must fit in GPU or CPU memory for latency, and edge deployments impose hard limits), so the community has treated memory as a first-class budget (the EfficientQA competition (Min et al., 2021) ran 6 GiB and 500 MiB tracks). (Our experiments use a more compact 384-dimensional FP16 setting.)

Two families of method reduce an index's footprint: *compression*, which spends fewer bits per document embedding (product quantization, scalar quantization, binary hashing), and *selection* (pruning), which retains fewer documents. The two are evaluated under *incompatible protocols*: compression over a fixed document set, varying the bits per vector; selection at a fixed *retention rate*, varying how many documents are kept. The question a storage-budgeted practitioner faces, which method makes the best use of a fixed budget, cannot be expressed under either protocol, and the gap is not cosmetic: methods at equal retention can differ in per-document storage by up to 16× (FP16 vs. binary), so equal-retention comparison answers a different question than a storage-bound deployment faces, and the two answers can disagree.

Placing the two families on a common per-document budget exposes the confound and then settles a question. On the same SciFact/MiniLM setting where prior work reports binary quantization as catastrophic at a fixed per-vector budget (Pati, 2025), a fair verdict for a fixed document set, that verdict *reverses* on the common storage axis, where binary's cheaper documents outrank full precision at every budget from 4 to 384 bits/document. The direction of the reversal is unsurprising (a cheaper code buys more documents); what matters is that the two protocols give *opposite* answers, so which one a practitioner measures is consequential when the deployment is storage-bound rather than vector-count-bound. With the axis fixed, the protocol settles the cross-family question, and the answer has two parts. At a fixed code, selection genuinely helps: the best off-the-shelf selector significantly beats random selection on all four core corpora (§5.4). But choosing a cheaper code is the larger lever: charged the same storage, the standard selectors the literature proposes almost never beat a trivially cheap quantization code, and the apparent cross-family value of selection is mostly absorbed by code choice.

This is not merely a fact about the instruments we test. A closed-form dominance condition (Eq. (4)) shows the disadvantage is structural: compression's cheaper code keeps more documents per bit, so to win, a selector must overcome that coverage advantage by document choice alone. That is a generalization problem, not a tuning one. A selector commits to its kept subset *before* seeing the test queries, so it must predict from training signal alone which documents future queries will need, and our own results show that prediction is hard: dense query-likelihood, the one selector we test built to do exactly this, fails to beat random selection across encoders (Section 5.5). Whether a *trained* selector could clear the bar is open; what we establish is that the bar is steep and that the obvious approach to clearing it underperforms.

Our contribution is the common-budget protocol and the two-part result it exposes, rather than a new method. Across nine BEIR corpora and three encoders, selection significantly beats compression on only one well-powered corpus, BM25 selection on a single financial corpus, and no measured corpus property (relevance density, intrinsic dimension, neighborhood similarity, query/document lexical overlap, or the BM25-versus-dense retrieval gap) predicts where selection helps (Section 5). *Storage-equivalent evaluation* fixes a per-document bit budget and lets each method retain $m = \lfloor BN/b \rfloor$ documents (b its bits-per-document, N the corpus size), placing pruning, quantization, and dimensionality reduction on one axis; every selector we study (random, k-center, k-means cluster centroid, and lexical and dense query-likelihood) is a standard, off-the-shelf instrument used to probe it. A PCA dimensionality-reduction case study on the same axis, dominated by quantization at every budget, shows the protocol ranks mechanistically different methods (Section 5.6).

2 Related Work

Dense retrieval compression. Product quantization (PQ) (Jégou et al., 2011) decomposes high-dimensional vectors into independently quantized subvectors; OPQ (Ge et al., 2014) and LOPQ (Kalantidis & Avrithis, 2014) learn rotations that reduce quantization error, learned codebooks add retrieval-aware objectives (Guo et al., 2020), and GPU-accelerated search makes these practical at billion scale (Johnson et al., 2021). Scalar quantization spans FP16 to INT8 and below; binary hashing (Wang et al., 2018) is the one-bit extreme, sharpened by RaBitQ's error-bounded codes (Gao & Long, 2024), and LeanVec (Tepper et al., 2024) combines dimensionality reduction with quantization and secondary-vector rescoring. Matryoshka Representation Learning (Kusupati et al., 2022) and ESE (Li et al., 2025) make dimensionality itself elastic, an orthogonal axis the framework could incorporate; late-interaction models (ColBERT (Khattab & Zaharia, 2020), ColBERTv2 (Santhanam et al., 2022)) trade higher per-document storage for finer matching, and MTEB (Muennighoff et al., 2022) standardizes encoder comparison. All of this work compresses a *fixed* document set; its interaction with document selection has received far less attention.

Corpus pruning and selection. Static index pruning removes postings by term-frequency or document-quality signals (Carmel et al., 2001; Büttcher & Clarke, 2006); query-likelihood pruning retains the documents most likely relevant to typical queries, with query-driven segment selection (Kim et al., 2021) and clustering-based selection its dense-era analogues (hard-negative mining (Zhan et al., 2021) and contrastive pretraining (Izacard et al., 2022) improve the encoder itself). These pruning studies vary *how many* documents are kept at a fixed per-vector precision, while compression work (including binary passage retrieval (Yamada et al., 2021)) varies the *precision* over a fixed set; the two are reported under separate protocols and never

compared head-to-head. Calls for cost-aware IR benchmarking (Santhanam et al., 2023), and evidence that strong-looking gains often shrink against well-tuned trivial baselines (Ferrari Dacrema et al., 2019; Yang et al., 2019), motivate both the common-budget protocol and the honest negative result it returns.

Memory-budget benchmarking. Standard IR evaluation (Recall@k, NDCG@k (Järvelin & Kekäläinen, 2002)) over BEIR's diverse domains (Thakur et al., 2021) reports effectiveness at fixed retention percentages, ignoring the differing storage cost of compression ratios. Plotting quality against memory is not new: ANN-Benchmarks (Aumüller et al., 2020) and the Big ANN Challenge (Simhadri et al., 2022) rank indexes under fixed hardware budgets, but over a *fixed document set*. Storage-equivalent evaluation differs in *what varies*: it lets the retained set vary so selection trades off directly against compression on one bits-per-document axis. This is not a reparameterization of recall-versus-memory curves: every point on a fixed-corpus curve shares the same document set, so such curves carry *no information* about selection by construction, and the cross-family comparison requires the coupling $m = \lfloor BN/b \rfloor$ those benchmarks cannot express. The closest prior work stays on a fixed corpus: Zouhar et al. (2022) and Pati (2025) compare precision against dimensionality reduction at fixed memory (the latter, on our SciFact/MiniLM setting, reports binary as catastrophic, the verdict that *inverts* on the common axis where binary's cheaper documents outrank FP16 at equal storage), and CoRECT (Caspari et al., 2026) evaluates eight compression methods to 100M passages but varies compression intensity, never the retained set. Two works combine the levers without a controlled axis: Izacard et al. (2020) mix dimension reduction, quantization, *and* passage filtering to hit a 6 GB *target* rather than deriving each method's affordable document count from a shared budget; and in LLM inference Zhang et al. (2025) trade token pruning against KV-cache quantization on a shared cache budget, the analogue we transfer from transient per-token state to the persistent document index. The gap is specific: compression studies vary bits-per-vector over a fixed corpus, pruning studies vary the retention ratio, and ANN memory benchmarks vary index memory over a fixed corpus; to our knowledge no prior work places document *selection* and *compression* on a common per-document bit budget with the retained set allowed to vary.

3 Problem Formulation

3.1 Setting

Consider a document corpus $\mathcal{D} = \{d_1, \ldots, d_N\}$ with corresponding dense embeddings $\mathcal{E} = \{e_1, \ldots, e_N\}$, where each $e_i \in \mathbb{R}^D$. In dense retrieval, given a query embedding $q \in \mathbb{R}^D$, we retrieve the top-k documents by inner product similarity: $\text{argmax}_{i \in [N]} \langle q, e_i \rangle$. Throughout, k denotes this retrieval cutoff (e.g. Recall@10) and m denotes the number of documents *retained* in the index; the two are unrelated. Our goal is to select and compress a subset of documents to fit within a storage budget B, while maximizing retrieval effectiveness on unseen queries. For a retained subset $\mathcal{S} \subseteq \mathcal{D}$, retrieval quality is measured by recall at cutoff k over that subset:

$$\text{Recall@}k(\mathcal{S}) = \frac{|\text{Relevant} \cap \text{Retrieved}_k(\mathcal{S})|}{|\text{Relevant}|}, \tag{1}$$

where $\text{Retrieved}_k(\mathcal{S})$ is the top-k result list computed over $\mathcal{S}$ under the method's stored representation. The two evaluation protocols below differ only in how $|\mathcal{S}|$ is fixed.

3.2 Storage-Equivalent Evaluation (Proposed)

Current practice fixes a retention rate $r \in (0, 1]$ and compares methods at equal $m = \lfloor rN \rfloor$ (Eq. (1)), implicitly assuming equal per-document storage, which is false: at 20% retention FP16 ($D = 384$) costs 16× more than binary ($0.2N \times 384 \times 16$ vs. $0.2N \times 384$ bits) yet is scored as equivalent. We propose instead fixing the storage budget B (in bits per document, averaged over the full corpus, i.e. total storage $B \times N$ bits) and allowing each method to retain as many documents as fit within this budget:

$$m_{\text{method}} = \left\lfloor \frac{B \times N}{b_{\text{method}}} \right\rfloor \tag{2}$$

Table 1: Dataset statistics. Relevance density is the fraction of documents relevant to at least one query; ceiling is the uncompressed upper bound (Recall@10 at 100% retention, FP16), the reference point for all budgeted results.

Dataset	Domain	Documents	Queries	Rel. density	Ceiling
SciFact	Scientific	5,183	300	5.5%	76.19%
FIQA	Financial	57,638	648	3.0%	44.25%
NFCorpus	Medical	3,633	323	86.1%	15.31%
NQ†	Wikipedia QA	500,000	3,452	0.8%	84.99%

†All NQ experiments use a fixed 500,000-document subsample of the full 2.68M-passage corpus (same subsample across methods and seeds) that *retains every judged-relevant document* plus a random draw of non-relevant passages, so the uncompressed ceiling is not depressed by subsampling (verified on the subset). Shrinking the distractor pool inflates *absolute* NQ recall identically for every method, so relative comparisons are unaffected and we compare within-corpus only.

where b_{method} is the bits-per-document for that compression method.[1] Storage-equivalent evaluation reports Eq. (1) at equal B, i.e. with $|\mathcal{S}| = m_{\text{method}}$ varying per method. This ensures all methods operate under the same storage constraint. The budget counts embedding bits (a flat index); Section 5 quantifies the document-identifier and ANN-structure overheads and shows every conclusion survives charging them, and Section 6.1 treats the text-store asymmetry and tiered (memory+disk) designs. Aggressive compression methods (e.g. binary) can retain more documents, potentially offsetting their per-document quality loss through greater coverage: on SciFact at $B = 4$ bits/document, the same 20,732-bit pool affords FP16 3 documents where binary affords 53 and PQ48×4 affords 107.

3.3 The Selection Problem

Given compression method c with bits-per-document b_c and storage budget B, we can retain $m_c = \lfloor BN/b_c \rfloor$ documents. The selection problem is to choose the subset of that size maximizing expected recall,

$$\mathcal{S}^* = \text{argmax}_{\mathcal{S} \subseteq \mathcal{D}, |\mathcal{S}| = m_c} \mathbb{E}_q[\text{Recall@}k(q; \mathcal{S})], \tag{3}$$

where the expectation is over the query distribution. This optimization is intractable without knowledge of future queries, motivating the heuristic selection strategies we evaluate in Section 4.

4 Experimental Setup

4.1 Datasets

We evaluate on four datasets from the BEIR benchmark (Thakur et al., 2021), chosen to span diverse domains and corpus structures (Table 1), and replicate the stress-budget comparison on five further BEIR corpora (Section 5.4). These four span a wide range of relevance densities (0.8% to 86.1%), corpus sizes (3,633 to 500,000 documents), and domain characteristics.

4.2 Embeddings

All documents and queries are encoded with `all-MiniLM-L6-v2` (Wang et al., 2020), a Sentence-Transformers (Reimers & Gurevych, 2019) checkpoint producing 384-dimensional embeddings normalized for cosine similarity. This model balances effectiveness and efficiency, making it representative of production

[1]Eq. (2) counts embedding bits only, i.e. flat-index storage; the overhead of an ANN structure layered on top (e.g. HNSW links) is likewise excluded here. Because every method discards documents at these budgets ($m < N$), each must also store a map from retained slot to original document identifier, costing $\lceil \log_2 N \rceil$ bits per retained document (quantization that keeps the full corpus in its original order would need none). This overhead is modest, e.g. ≈27 bits against a 384-bit binary embedding at $N{=}10^8$ (∼7%), and shrinks each method's retained set by the method-dependent factor $b/(b + \lceil \log_2 N \rceil)$, about 7% for binary versus under 1% for FP16; it narrows but does not reverse binary's coverage advantage, so we omit it from the budget.

Table 2: Compression methods and storage costs.

Method	Bits/dimension	Bits/document (D=384)
FP16	16	6,144
INT8	8	3,072
SQ4	4	1,536
PQ48×8	–	384
PQ96×4	–	384
PQ48×4	–	192
PQ24×4	–	96
Binary	1	384

deployments. Section 5.5 replicates the headline experiments on two further encoders (BGE-small-en-v1.5 and the 1024-dimensional BGE-large-en-v1.5).

4.3 Compression Methods

We evaluate eight compression methods spanning the precision spectrum (Table 2). Retrieval fidelity is real wherever it could matter: binary is retrieved with `IndexBinaryFlat` (symmetric Hamming search over binarized queries and documents, conservative *against* binary, since asymmetric search with full-precision queries would only raise its scores), all PQ variants with real `IndexPQ` codes, and INT8/SQ4 with real Faiss scalar quantizers (`IndexScalarQuantizer`); FP16 is modeled as lossless full-precision retrieval with its 16-bit storage charged to the budget. (Full-precision retrieval over the same retained sets reproduces the INT8/SQ4 values to within 0.02 points at the stress budgets.)

Product quantization (PQ48×8 and PQ48×4) divides the 384-dimensional space into 48 subspaces, each quantized to 8 or 4 bits respectively. All indexing uses the Faiss library (Douze et al., 2024).

4.4 Storage Budgets

We evaluate three nested budget ranges, all in bits/document averaged over the full corpus. The *stress-test* range comprises seven budgets (1, 2, 4, 6, 8, 12, and 16 bits/document), with 4 bits/document as the primary diagnostic point; the *deployment* range extends to 32, 64, and 128 bits/document (Tables 3 and 7); and the ranking-reversal sweep additionally evaluates up to 384 bits/document, the budget at which binary stores the full corpus (Figure 1, Table 6).

All methods of Table 2 are run with real quantized retrieval across budgets 4–128 bits/document under random selection; the headline tables report binary and PQ48×4 because the rest of the ordering is mechanical, in exactly the direction the dominance condition (§5.2) predicts: at every sub-saturation budget the methods rank in inverse bits-per-document order, on all four corpora (at 4 bits/document on SciFact: FP16 0.08%, INT8 0.20%, SQ4 0.31%, binary 1.07%, PQ96×4 1.12%, PQ48×4 1.98%, PQ24×4 3.72%; at 64 bits/doc PQ24×4 reaches 42.1% vs. PQ48×4's 26.7%). The lone exception is the theory's other regime: on NFCorpus at 128 bits/document PQ24×4 already stores the *full* corpus, so extra budget buys it nothing and fidelity finally decides (7.45% vs. 8.71%), consistent with Eq. (4)'s sub-saturation qualifier.

Concretely (storage = bits/doc $\times N/8$ bytes), a 100-million-document corpus costs ≈76.8 GB at FP16 ($16 \times 384 = 6{,}144$ bits/document) and ≈4.8 GB at binary (384), while the 4 bits/document budget is only ≈50 MB, a 1,536× reduction from FP16; no single compression method reaches that budget without also discarding documents, which is exactly why the framework must let the retained set vary.

4.5 Selection Methods

We compare five standard document-selection methods, all widely used and parameter-light:

- **Random**: uniform random selection.
- **k-Center (diversity)**: greedy farthest-point selection of documents maximally distant from those already chosen, maximizing embedding-space coverage.
- **Cluster Centroid**: k-means clustering, retaining the document nearest each centroid, ensuring representation of each topic cluster.
- **Query-Likelihood (BM25)**: retain documents with the highest average BM25 score across the traffic-split queries, via Pyserini (Lin et al., 2021); this captures which documents are most likely relevant to typical queries (Carmel et al., 2001).
- **Dense Query-Likelihood**: score each document by its mean cosine similarity to the traffic-split query embeddings (same encoder); a stronger, embedding-based use of the traffic signal than BM25's lexical match.

Selectors store at binary compression (384 bits/document) for same-retention comparisons at a fixed budget; to separate selection from code choice we additionally evaluate them storing at the cheaper PQ48×4 code (192 bits/document) at the stress budget (§5.4).

4.6 Evaluation Protocol

To prevent information leakage, we split queries per seed into three disjoint sets: **Traffic** (50%), which drives the query-likelihood selectors (BM25, dense query-likelihood); **Validation** (25%), held out; and **Test** (25%), used only for final evaluation and never seen during selection. The unsupervised selectors (random, k-center, cluster centroid) use no queries, and the query-likelihood selectors see only the traffic split, so test queries never influence which documents are retained. After the split the test set holds 75, 162, 82, and 863 queries on SciFact, FIQA, NFCorpus, and NQ, and 352, 250, 13, 13, and 2,500 on the five replication corpora (ArguAna, SciDocs, TREC-COVID, Webis-Touché2020, Quora); the smaller counts, most acutely TREC-COVID and Webis-Touché2020 at 13 test queries each, leave several per-corpus comparisons low-powered, and we read their point estimates accordingly throughout the results.

All results report mean Recall@10 across 25 random seeds (the seed controls the traffic/validation/test split), except where a smaller count is noted (the reranking study, and NQ in the selector+PQ48×4 cells, at 5 seeds). Significance is assessed via paired two-sided t-tests over per-seed differences (Smucker et al., 2007), with paired Cohen's d and Holm–Bonferroni correction within each comparison family. We test at the seed level rather than the IR-conventional per-query level because the dominant noise source here is the split re-draw itself (it changes both the traffic-driven selector scores and the retained set), and per-query observations are not comparable across seeds, which retrieve over different retained indexes. Standard deviations accompany every headline comparison. All code, dataset configurations, and a deterministic script that regenerates every embedding from the public BEIR corpora and public encoder checkpoints will be released upon acceptance; we omit the repository link here to preserve anonymity.

5 Results

Charged equal per-document storage, selection and compression split the verdict. At a fixed code, selection helps: the best selector significantly beats random on all four core corpora, at both binary and PQ48×4 (§5.4). Across codes, compression is the larger lever: at deployment budgets a cheap product-quantization code beats every selector measured there (§5.3), and at the storage-scarce stress budget, among the corpora with enough test queries to resolve the comparison, selection significantly wins on exactly one, BM25 selection on FIQA. No measured corpus property predicts *where* it helps: five properties, including a full-corpus BM25-versus-dense gap built to capture lexical-distinctiveness, fail to order the corpora (§5.4). We start with the diagnostic that motivates the protocol (§5.1), state the mechanism as a budget-free dominance condition bounding *any* selector (§5.2), then turn to the selection results.

5.1 The Ranking Reversal

Equal-retention comparison scores methods as equivalent while their per-document costs differ by up to 16×; charging the cost reverses the ranking. That reversal is the diagnostic that motivates the protocol, not a finding in its own right. Consider SciFact. Under retention-based evaluation at 1% retention, FP16 and binary keep the same 51 documents and perform comparably (1.12% vs. 1.07% Recall@10 over 25 seeds): at such low retention, whether a relevant document is in the kept set matters far more than embedding precision. But they are not equivalent in cost: FP16 consumes $51 \times 384 \times 16 = 313$K bits to binary's 20K, a 16× difference the comparison ignores. Under storage-equivalent evaluation at 4 bits/document (binary's storage), that 16× buys binary far more documents: FP16 retains only $\lfloor 4 \times 5183/6144 \rfloor = 3$ documents (near-zero recall) while binary retains 53 (1.07%). The pattern is uniform across all four corpora: FP16 and binary tie at 1% retention, but at 4 bits/document FP16 collapses to 0.06–0.08% Recall@10 everywhere while binary is unchanged.

The reversal at moderate retention. At 1% retention FP16 and binary tie, so the storage axis merely separates them. A true reversal needs the equal-retention *winner* to lose at equal storage, and that takes higher retention, where per-vector fidelity has room to matter. Figure 1 sweeps random selection from 1% to 100% retention for FP16 and binary on SciFact, NFCorpus, and NQ (25 seeds). At equal retention, FP16 ranks strictly and significantly above binary once enough documents are kept: on SciFact it leads 17.2% to 15.4% Recall@10 at 20% retention ($p < 0.001$), and on NFCorpus and NQ it leads at every rate ≥10%. At equal storage the ordering inverts at every budget from 4 to 384 bits/document (Figure 1b), with a roughly constant margin (∼12× on SciFact, ∼6–7× on NFCorpus, ∼13–14× on NQ) rather than a spike at the saturating endpoint; Table 6 (Appendix B) traces the SciFact curve concretely.

The reversal is not specific to the MiniLM encoder: it reproduces on bge-small-en-v1.5 (384-d) and bge-large-en-v1.5 (1024-d) (Xiao et al., 2023) in all 24 (encoder×corpus×budget) cells, the margin if anything larger at higher dimension (Appendix A).

5.2 The Mechanism: A Budget-Free Dominance Condition

Why does coverage dominate per-vector fidelity so reliably that no selector we test closes the gap? A closed-form dominance condition answers this, and as a by-product predicts the equal-storage winner from two inputs that are cheap to measure, a fitted recall–retention curve and each method's bits per document, without running a budget sweep.

A closed-form crossover predictor. On all three corpora the recall–retention curve is captured by a two-parameter law $f(r) = a\, r^p$ (a the full-corpus recall, p a curvature exponent), fit on five retention points (1, 10, 20, 50, and 100%) at $R^2 > 0.999$. The fit generalizes rather than merely interpolating its own anchors. On SciFact a 64 bits/document budget lets binary retain 16.7% of the corpus, a retention rate not among the five fitted; the law predicts 12.98% Recall@10 there, and the measurement is 12.84% (NFCorpus and NQ agree within a few tenths; Appendix B). The curve then becomes an equal-storage predictor: a method costing b bits/document retains the fraction $\min(1, B/b)$ of the corpus at budget B, so its recall is $R(B) = f(\min(1, B/b))$. Solving $R_{\text{fp16}}(B) = R_{\text{bin}}(B)$ gives the crossover budget B^* at which the two methods would tie, and on every corpus B^* lands far above binary's 384 bits/document saturation point: the formula recovers analytically what the sweep shows empirically.

A budget-free dominance condition. The predictor sharpens into a budget-free condition. Write each method's sub-saturation recall under the fitted law as $R_i(B) = a_i\,(B/b_i)^{p_i}$, valid for $B \le b_i$. *If the exponents are shared ($p_i = p$), the cheaper code c dominates the more expensive code e at every budget where both are sub-saturated if and only if*

$$\frac{a_c}{a_e} > \left(\frac{b_c}{b_e}\right)^p, \tag{4}$$

a condition independent of the budget (the recall ratio R_c/R_e is then constant in B; with distinct exponents it is monotone, so at most one interior crossover exists). The reading: a_c/a_e is the fidelity the cheap code retains at full corpus, $(b_c/b_e)^p$ the recall value of the coverage it would forgo. For binary vs. FP16 the condition holds with margins of 12.0×, 5.9×, and 12.1× on SciFact/NFCorpus/NQ (a-ratios 0.91/0.86/0.93

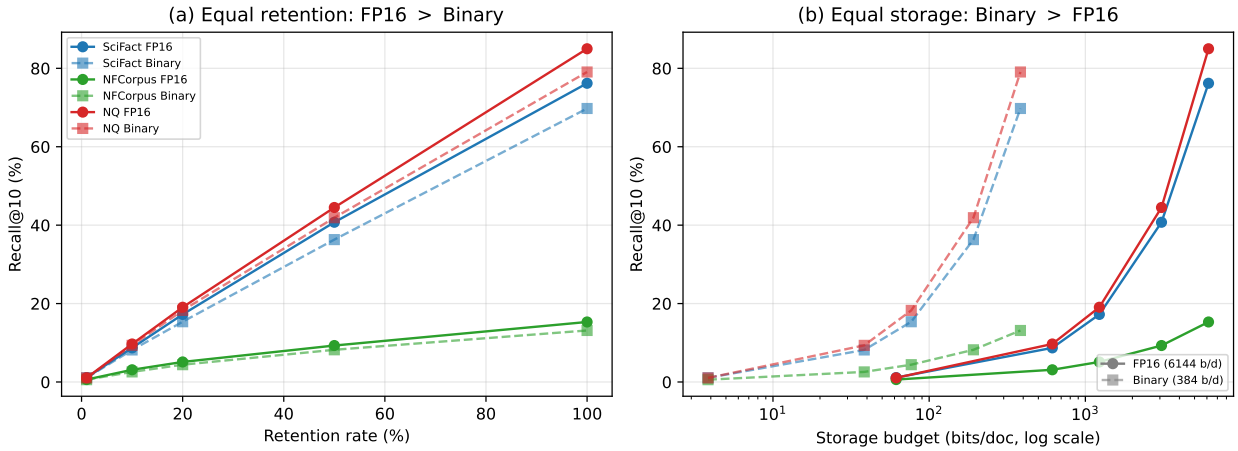

Figure 1: The ranking reversal: the method that ranks best at equal retention can rank worst at equal storage, so which compression method is best has no budget-free answer. **(a)** At equal retention, FP16 (solid) outranks binary (dashed) on all three corpora once enough documents are kept. **(b)** At equal storage (bits/document, log scale), binary outranks FP16, because its 16× lower per-document cost buys 16× the documents; storage-axis points are *interpolated* from each method's recall–retention sweep at the retention its budget affords. Same two methods, same corpora, opposite ordering.

against thresholds 0.076/0.147/0.077): binary would need to destroy over 85% of full-corpus recall before 16× coverage failed to pay for it; it destroys under 14%. With method-specific exponents (which differ by ≤ 0.017) the interior crossover lands at absurd scales (10^{-61} to 10^{195} bits/document): no crossover exists at any realizable budget. The condition's practical bite is at saturation boundaries and between nearby codes; its sub-saturation qualifier identifies where the cheap-code rule fails, and the one exception in the broadened code grid of §4 (NFCorpus at 128 bits/document, where PQ24×4 has saturated) falls at such a boundary, consistent with it. We read the condition as a mechanism, not a flip-predictor, and do not rest on its calls at individual boundaries.

The reversal survives realistic storage accounting. Eq. (4) also quantifies the overheads the budget footnote of §3 identifies: the document-identifier map (≈27 bits at $N = 10^8$) barely moves the threshold, and even HNSW-style links (≈1,024 bits per retained document, M=32 neighbors at 32-bit ids), which compress the effective FP16-to-binary cost ratio from 16× to ≈5.1×, leave dominance holding with margins of 4.1×, 2.6×, and 4.1×, the adjusted crossover rising only to 5,800–6,700 bits/document: the reversal is not an artifact of flat-index accounting.

Which *selection* method wins at equal storage is a separate question, and the answer, taken up next, is that selection rarely wins at all, and that no corpus property predicts when it does.

5.3 The Practical Regime: Moderate Storage Budgets

At deployment-relevant budgets (64 and 128 bits/document), Table 3 reports each selector storing at binary against PQ48×4 with random selection. The finding is unambiguous: **code choice is first-order**. PQ48×4+random reaches Recall@10 of up to 50–54% and beats every binary selector on every corpus; switching binary→PQ48×4 at random selection gains up to 2.0× (SciFact, 24.66 → 50.04 at 128 bits/document). Among the binary selectors, cluster centroid helps on the general-domain NQ corpus (19.41/32.22 vs. random's 15.38/29.14) and dense query-likelihood helps slightly on SciFact (14.10/26.31), but none approaches the cheaper code. Choose the cheapest code that has not saturated the corpus, first.

Table 3: Recall@10 (%) at the deployment budgets (64/128 bits/document, 25 seeds). Each selector stores at binary; the last row is PQ48×4 with random selection. Best per column in bold. The cheaper PQ48×4 code beats every binary selector on every corpus.

	SciFact		**FIQA**		**NFCorpus**		**NQ**	
Method	64	128	64	128	64	128	64	128
Binary + Random	12.84	24.66	9.96	17.52	3.78	6.22	15.38	29.14
Binary + Cluster Centroid	12.12	25.53	9.00	15.58	3.53	6.28	19.41	32.22
Binary + Dense Query-Likelihood	14.10	26.31	7.97	17.09	4.18	6.21	15.95	30.19
PQ48×4 + Random	**26.70**	**50.04**	**17.40**	**26.51**	**5.74**	**8.71**	**28.91**	**53.95**

Table 4: Recall@10 (%) at 4 bits/document (25 seeds). All selectors store at binary; the middle block gives the two compression references. Best per column (among all rows) in bold. The verdict row is the best selector vs. PQ48×4+random (paired t-test; * = $p < 0.05$). NQ uses the 500K subsample (Table 1 footnote): its *absolute* recall is inflated relative to full-corpus NQ and not comparable to published full-corpus numbers; we read NQ within-corpus only.

Method	SciFact	FIQA	NFCorpus	NQ
Random	1.07	0.79	0.60	1.08
k-Center	0.47	0.55	0.46	1.07
Cluster Centroid	1.29	1.08	0.61	1.93
Query-Likelihood (BM25)	0.51	**2.08**	0.50	1.79
Dense Query-Likelihood	**2.12**	0.49	0.89	0.98
Binary + Random	1.07	0.79	0.60	1.08
PQ48×4 + Random	1.98	1.49	**0.90**	**2.04**
Best selector vs. PQ48×4	1.07×	1.40×*	0.99×	0.95×
(p)	(0.66)	(**0.003**)	(0.97)	(0.38)

5.4 The Stress Budget: Where, If Anywhere, Selection Helps

At the 4 bits/document stress budget, where every method retains only ∼1% of documents (tens of documents on the smaller corpora), the picture is more nuanced. Recall@10 stays well-defined at this scale: it asks whether a query's relevant documents both survive selection and rank in the top 10 of what is retrieved, so a query whose relevant document was pruned scores 0, and the metric is dominated by whether selection kept the right documents, which is exactly the quantity under test. Table 4 compares the five selectors (all storing at binary) against binary+random and PQ48×4+random. Against binary+random, the same code the selectors store at, the best selector wins significantly on all four corpora (paired t-test, 25 seeds; Cohen's d from 0.45 to 1.56). Against the stronger PQ48×4 reference, only *one* corpus shows a significant selection win: **FIQA**, where BM25 selection (2.08%) beats PQ48×4+random (1.49%) by 1.40× ($p = 0.0025$, $d = 0.68$). On SciFact, NFCorpus, and NQ the best selector and PQ48×4 are statistically indistinguishable (PQ numerically ahead on NFCorpus and NQ). This verdict is cross-code, though: it pits selection at binary against random at the cheaper PQ48×4. Storing the selectors *at* PQ48×4 instead, so they keep the same *number* of documents as random, the best selector significantly beats random+PQ48×4 on all four corpora (SciFact 3.64 vs. 1.98, FIQA 3.39 vs. 1.49, NFCorpus 1.36 vs. 0.90, NQ 3.87 vs. 1.99; 25 seeds, 5 on NQ; ratios 1.5–2.3×, all $p < 0.01$): selection helps at a *fixed* code, and the negative result is comparative, a cheaper code being the larger lever. What that single cross-code win means, and whether any corpus property predicts it, we take up next.

Replication, and the limits of prediction. We ran the same 4 bits/document comparison on five further BEIR corpora. By point estimate the best selector beats PQ48×4+random on three (SciDocs 1.44 vs. 1.35,

TREC-COVID 0.67 vs. 0.32, Webis-Touché2020 2.83 vs. 1.05) and loses on two (ArguAna, Quora). But two of the three apparent wins, TREC-COVID and Webis-Touché2020, retain only 13 test queries each after the split, so their point estimates are near noise; restricting to corpora with enough test queries to resolve the comparison leaves *exactly one* significant selection win in the entire study, FIQA. It is tempting to read a pattern into the wins, that lexical matching helps on keyword-distinctive corpora, so we tested it directly. We computed five corpus properties, relevance density, intrinsic dimension, neighborhood similarity, query/document lexical overlap, and the full-corpus BM25-versus-dense retrieval gap, and asked whether any orders the nine corpora by their selection-vs-compression ratio. None does. The most lexically-motivated property, the BM25-versus-dense gap (which, unlike raw overlap, controls for whether a corpus is simply easy), is uncorrelated among the seven well-powered corpora (Spearman $\rho = 0.21$, $p = 0.64$); its only two lexical-favoring values fall on the two 13-query corpora; and FIQA, the one significant selection win, is among the *most dense-favoring* corpora we measured: dense retrieval beats BM25 there by 14 points. So the single method that beats compression, BM25 used *as a selector*, wins on a corpus where BM25 used *as a retriever* loses to dense by 14 points; the win is therefore not a lexical-distinctiveness effect, and no measured property explains it. We report no predictor of where selection helps: the effect is real on FIQA, rare elsewhere, and uncharacterized, the lexical story included. This is necessarily an underpowered null, with a single well-powered positive instance there is nothing to correlate against, so the honest claim is that the properties we tried do not predict the one win, not that none could.

5.5 Generalization Across Embedding Models

To check that the fragility of embedding-based selection is not specific to MiniLM, we repeat the 4 bits/document dense query-likelihood-vs-random comparison on two further encoders, bge-small-en-v1.5 (384-d) and bge-large-en-v1.5 (1024-d) (Xiao et al., 2023), on the three corpora for which all encoders' embeddings were available (Table 8; binary storage, so the 1024-d model affords 2.67× fewer documents at a given budget, and we charge that correctly). The dense-QL selector shows *no consistent, encoder-robust advantage* over random selection: across the nine (model × corpus) cells, after Holm–Bonferroni correction over the nine comparisons, it significantly beats random on only two (MiniLM/SciFact and bge-large/FIQA), and these are *different* corpora. Two further cells are nominally significant before correction but point in *opposite* directions (dense-QL ahead on MiniLM/NFCorpus, behind on MiniLM/FIQA), so even uncorrected the selector has no consistent sign; the SciFact edge seen on MiniLM is non-significant on bge-small ($p = 0.09$) and reverses on bge-large ($p = 0.29$). Embedding-based selection is thus fragile across encoders as well as corpora, reinforcing compression as the reliable default; the one selector that significantly beat compression, BM25, is by construction independent of the embedding model (which is also why its single win is not explained by any embedding-based corpus property).

The ordering is not an artifact of the shallow cutoff. Reranking the top-100 candidates from each budgeted index with an off-the-shelf cross-encoder (`ms-marco-MiniLM-L-6-v2`) preserves the code-choice ordering (PQ48×4 beats binary at every budget on post-rerank nDCG@10), so the conclusion survives a realistic retrieve-then-rerank pipeline rather than being an artifact of measuring a budgeted first stage at rank 10 (full numbers, SciFact and NFCorpus at 4/32/128 bits, in Appendix C).

5.6 Heterogeneous Methods on the Same Axis

The framework ranks *mechanistically different* methods, not just pruning against quantization. Placing PCA dimensionality reduction on the common axis (a projection to d components at FP16 costs $16d$ bits/document), it is *dominated by product quantization at every budget on every corpus*, and the validation-selected dimension always lands at the small end of the sweep ($d = 16$ or 32): coverage beats per-vector fidelity again. Neither an optimized quantizer (OPQ tracks plain PQ48×4) nor a data-oblivious Johnson–Lindenstrauss projection (worse than PCA) changes the verdict (Appendix D).

6 Discussion

Practical guidance. Choose the cheapest code that has not saturated the corpus first: at deployment budgets a product-quantization code (PQ48×4, or PQ24×4 in the broadened grid) beats every binary selector on every corpus we measured, so code choice is first-order. Selection is rarely worth adding, it beats the cheapest non-saturated code on one corpus only (lexical BM25 on FIQA), and no measured corpus property predicts where it will help, so the rule for selection is empirical, not a priori: run a validation-split pilot at the operating budget and adopt selection only if it beats compression there.

6.1 Why a Single Storage Axis

We make storage the first evaluation axis because it is hardware-independent and the only axis on which pruning, quantization, and dimensionality reduction reduce to a common quantity, bits per document; query-time latency depends on the approximate-nearest-neighbor structure layered on top and the hardware under it. Latency is the natural orthogonal second axis, and one coarse proxy is already explicit in our accounting: the retained count $m = \lfloor BN/b \rfloor$ is exactly the first-stage candidate set, so binary's equal-storage win is bought with up to 16× as many candidates as FP16 (though each comparison is a far cheaper Hamming popcount, so the wall-clock penalty is well below 16×). Two indexes of equal storage can still differ substantially in latency, build cost, and memory bandwidth; characterizing that second axis is future work.

The budget also charges the embedding *index* only, not the document text, and the asymmetry cuts both ways: where raw text dominates total storage, pruning also shrinks the text store (a saving compression cannot claim), so index-only accounting is *conservative against selection*; conversely, a system that must keep every document servable cannot prune at all, whatever the index budget says. Recall already prices the irrecoverability of pruned documents, since relevant documents outside the retained set count as misses. Tiered designs pose no obstacle: DiskANN (Jayaram Subramanya et al., 2019) keeps compressed codes in memory and full-precision vectors on SSD, LeanVec (Tepper et al., 2024) rescores with secondary vectors, and each tier has a well-defined bits-per-document, so the protocol applies by fixing whichever tier's budget binds (typically memory); we study flat single-tier indexes to isolate the selection-versus-compression trade-off from tiering policy.

Outlook: from uniform to query-aware bit allocation. The protocol collapses selection and compression into one quantity, bits per document: selection is compression to zero bits, and this paper studies only the *uniform* allocation of the $B{\cdot}N$-bit budget. The natural generalization is *non-uniform, query-aware* allocation, a mixed-precision index spending more bits on documents likely to be queried and fewer (or zero) on the long tail: the dense-retrieval analogue of the token-versus-precision trade-off of Zhang et al. (2025), with Matryoshka embeddings (Kusupati et al., 2022) as a learned per-document precision knob. Whether query-aware mixed precision beats uniform compression is genuinely open: our result that query-likelihood *selection* rarely helps is a caution, but it probes only the zero-bits extreme of the allocation and leaves the interior unexplored.

6.2 Limitations

Small test sets and statistical power. After splitting, several datasets have few test queries (§4: SciFact 75, NFCorpus 82, and TREC-COVID and Webis-Touché2020 only 13 each), inflating variance, so several selection-versus-compression differences sit within seed noise (Section 5); we report effect sizes and p-values throughout.

Selector×code grid partially measured. The full eight-code grid is measured under random selection (§4); the non-random selectors are measured at two codes, binary and the cheaper PQ48×4 (§5.4), and the deployment-budget comparison (Table 3) covers cluster centroid and dense query-likelihood, with BM25 and k-center measured at the stress budget only. The binary→PQ48×4 pairing preserves the selection-beats-random ordering on all four core corpora and behaves as the documents-retained arithmetic predicts, so we expect no qualitative change across the remaining cells, but they are unmeasured.

Selectors tested, not all. We evaluate query-likelihood selectors (lexical BM25 and dense) alongside the classical baselines (random, k-center, cluster centroid); a learned or supervised pruner trained on the traffic split might behave differently and remains untested. Our claims concern document selection at these budgets, not any single instrument.

The crossover predictor is empirical. The dominance condition rests on the two-parameter law $f(r) = a\,r^p$. We rest the case on held-out validation, prediction of retention points *outside* the fit to within a few tenths of a point (Appendix B), rather than on the in-sample $R^2 > 0.999$, which on five points is weak evidence. The functional form is empirical, not derived from corpus geometry; a first-principles account of why the power law holds is future work, and the predictor should be read as a well-validated regularity, not a theorem.

Encoder scale. The reversal and selection results are measured on 384-d and 1024-d encoders (MiniLM, bge-small, bge-large); confirming them on larger, higher-dimensional embedders (e.g. 7B-parameter, 4096-d models, which require GPU encoding) is left to future work.

Broader Impact Statement

This work studies storage-efficient evaluation for dense retrieval; it proposes an evaluation protocol and analysis rather than a deployed system, and we are not aware of negative societal impacts specific to it. By clarifying when aggressive compression suffices, the framework may reduce the memory and energy footprint of retrieval indexes.

References

M. Aumüller, E. Bernhardsson, and A. Faithfull. ANN-Benchmarks: A benchmarking tool for approximate nearest neighbor algorithms. *Information Systems*, 87:101374, 2020.

S. Büttcher and C. L. A. Clarke. A document-centric approach to static index pruning in text retrieval systems. In *Proc. ACM CIKM*, pp. 182–189, 2006.

D. Carmel, D. Cohen, R. Fagin, E. Farchi, M. Herscovici, Y. S. Maarek, and A. Soffer. Static index pruning for information retrieval systems. In *Proc. ACM SIGIR*, pp. 43–50, 2001.

L. Caspari, M. Dinzinger, K. Ghosh Dastidar, C. Fellicious, J. Mitrović, and M. Granitzer. CoRECT: A framework for evaluating embedding compression techniques at scale. In *Proc. Eur. Conf. Inf. Retrieval (ECIR)*, 2026.

M. Douze, A. Guzhva, C. Deng, J. Johnson, G. Szilvasy, P.-E. Mazaré, M. Lomeli, L. Hosseini, and H. Jégou. The Faiss library. *arXiv preprint arXiv:2401.08281*, 2024.

M. Ferrari Dacrema, P. Cremonesi, and D. Jannach. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In *Proc. ACM RecSys*, pp. 101–109, 2019.

J. Gao and C. Long. RaBitQ: Quantizing high-dimensional vectors with a theoretical error bound for approximate nearest neighbor search. *Proc. ACM Manag. Data*, 2(3 (SIGMOD), Art. 167), 2024.

T. Ge, K. He, Q. Ke, and J. Sun. Optimized product quantization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(4):744–755, 2014.

R. Guo et al. Accelerating large-scale inference with anisotropic vector quantization. In *Proc. ICML*, pp. 3887–3896, 2020.

G. Izacard, F. Petroni, L. Hosseini, N. De Cao, S. Riedel, and E. Grave. A memory efficient baseline for open domain question answering. *arXiv preprint arXiv:2012.15156*, 2020.

G. Izacard, M. Caron, L. Hosseini, S. Riedel, P. Bojanowski, A. Joulin, and E. Grave. Unsupervised dense information retrieval with contrastive learning. *Trans. Mach. Learn. Res.*, 2022.

K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.

S. Jayaram Subramanya, F. Devvrit, H. V. Simhadri, R. Krishnaswamy, and R. Kadekodi. DiskANN: Fast accurate billion-point nearest neighbor search on a single node. In *Proc. NeurIPS*, 2019.

H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(1):117–128, 2011.

J. Johnson, M. Douze, and H. Jégou. Billion-scale similarity search with GPUs. *IEEE Trans. Big Data*, 7 (3):535–547, 2021.

W. B. Johnson and J. Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary Mathematics*, 26:189–206, 1984.

Y. Kalantidis and Y. Avrithis. Locally optimized product quantization for approximate nearest neighbor search. In *Proc. IEEE CVPR*, pp. 2321–2328, 2014.

V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-T. Yih. Dense passage retrieval for open-domain question answering. In *Proc. EMNLP*, pp. 6769–6781, 2020.

O. Khattab and M. Zaharia. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *Proc. ACM SIGIR*, pp. 39–48, 2020.

Y. Kim, R. Rahimi, H. Bonab, and J. Allan. Query-driven segment selection for ranking long documents. In *Proc. ACM CIKM*, pp. 3147–3151, 2021.

A. Kusupati et al. Matryoshka representation learning. In *Proc. NeurIPS*, 2022.

P. Lewis et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proc. NeurIPS*, 2020.

X. Li, Z. Li, J. Li, H. Xie, and Q. Li. ESE: Espresso sentence embeddings. In *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2025.

J. Lin, X. Ma, S.-C. Lin, J.-H. Yang, R. Pradeep, and R. Nogueira. Pyserini: A Python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proc. ACM SIGIR*, pp. 2356–2362, 2021.

S. Min et al. NeurIPS 2020 EfficientQA competition: Systems, analyses and lessons learned. In *Proc. Mach. Learn. Res. (NeurIPS Competition Track)*, volume 133, pp. 86–111, 2021.

N. Muennighoff, N. Tazi, L. Magne, and N. Reimers. MTEB: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*, 2022.

S. Pati. Dimension vs. precision: A comparative analysis of autoencoders and quantization for efficient vector retrieval on BEIR SciFact. *arXiv preprint arXiv:2511.13057*, 2025.

N. Reimers and I. Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proc. EMNLP-IJCNLP*, pp. 3982–3992, 2019.

K. Santhanam, O. Khattab, J. Saad-Falcon, C. Potts, and M. Zaharia. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In *Proc. NAACL*, pp. 3715–3734, 2022.

K. Santhanam, J. Saad-Falcon, M. Franz, O. Khattab, A. Sil, R. Florian, M. A. Sultan, S. Roukos, M. Zaharia, and C. Potts. Moving beyond downstream task accuracy for information retrieval benchmarking. In *Findings of the Assoc. Comput. Linguistics (ACL)*, pp. 11613–11628, 2023.

H. V. Simhadri et al. Results of the NeurIPS'21 challenge on billion-scale approximate nearest neighbor search. In *Proc. NeurIPS Competitions and Demonstrations Track*, volume 176 of *PMLR*, pp. 177–189, 2022.

M. D. Smucker, J. Allan, and B. Carterette. A comparison of statistical significance tests for information retrieval evaluation. In *Proc. ACM CIKM*, pp. 623–632, 2007.

M. Tepper, I. S. Bhati, C. Aguerrebere, M. Hildebrand, and T. Willke. LeanVec: Searching vectors faster by making them fit. *Trans. Mach. Learn. Res.*, 2024.

N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, and I. Gurevych. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Proc. NeurIPS*, 2021.

J. Wang, T. Zhang, et al. A survey on learning to hash. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(4): 769–790, 2018.

W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Proc. NeurIPS*, volume 33, pp. 5776–5788, 2020.

S. Xiao, Z. Liu, P. Zhang, and N. Muennighoff. C-Pack: Packaged resources to advance general Chinese embedding. *arXiv preprint arXiv:2309.07597*, 2023.

L. Xiong, C. Xiong, Y. Li, K.-F. Tang, J. Liu, P. N. Bennett, J. Ahmed, and A. Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *Proc. ICLR*, 2021.

I. Yamada, A. Asai, and H. Hajishirzi. Efficient passage retrieval with hashing for open-domain question answering. In *Proc. ACL-IJCNLP (Short Papers)*, pp. 979–986, 2021.

W. Yang, K. Lu, P. Yang, and J. Lin. Critically examining the 'neural hype': Weak baselines and the additivity of effectiveness gains from neural ranking models. In *Proc. ACM SIGIR*, pp. 1129–1132, 2019.

J. Zhan, J. Mao, Y. Liu, J. Guo, M. Zhang, and S. Ma. Optimizing dense retrieval model training with hard negatives. In *Proc. ACM SIGIR*, pp. 1503–1512, 2021.

J. Zhang, D. Zhu, Y. Song, W. Wu, C. Kuang, X. Li, L. Shang, Q. Liu, and S. Li. More tokens, lower precision: Towards the optimal token-precision trade-off in KV cache compression. In *Findings of the Assoc. Comput. Linguistics: EMNLP*, pp. 8092–8105, 2025.

V. Zouhar, M. Mosbach, M. Zhang, and D. Klakow. Knowledge base index compression via dimensionality and precision reduction. In *Proc. 1st Workshop on Semiparametric Methods in NLP (Spa-NLP), ACL 2022*, 2022. arXiv:2204.02906.

A Cross-Model Reversal

The equal-storage reversal of §5.1 uses MiniLM; it is not specific to that encoder. Repeating the direct-measurement sweep on bge-small-en-v1.5 (384-d) and bge-large-en-v1.5 (1024-d) (Xiao et al., 2023), on the three corpora for which all encoders' embeddings are available, reproduces it in every cell: binary outranks FP16 at all four budgets on both encoders (24 of 24 cells), and the margin is, if anything, larger on the higher-dimensional bge-large (Table 5; on SciFact at the binary-saturation budget, binary's advantage is 11.3× on bge-small and 12.9× on bge-large). The mechanism is unchanged: binary's 16× lower per-document cost buys 16× the documents, and at equal storage that coverage outweighs FP16's per-vector fidelity.

B Crossover Predictor Details

The closed form follows from the fitted law $f(r) = a\,r^p$: for budgets above binary's saturation point ($B \geq 384$ bits/document), binary stores the full corpus and its recall is pinned at its ceiling a_{bin}, so the crossover is where FP16's curve reaches that ceiling, $a_{\text{fp16}}(B/b_{\text{fp16}})^{p_{\text{fp16}}} = a_{\text{bin}}$, giving $B^* = b_{\text{fp16}}\,(a_{\text{bin}}/a_{\text{fp16}})^{1/p_{\text{fp16}}}$. The saturation assumption is self-consistent: every fitted B^* lands far above 384 bits/document (if a crossover existed below saturation, solving the two unsaturated power laws instead would expose it). B^* evaluates to 5568, 4965, and 5687 bits/document on SciFact, NFCorpus, and NQ, confirming there is no crossover

Table 5: The equal-storage reversal is encoder-robust: Recall@10 (%) at each encoder's binary-saturation budget $B = D$ bits/document, where binary stores the full corpus and FP16 retains only $\lfloor N/16 \rfloor$ documents (random selection, 25 seeds). Binary (bold) outranks FP16 on every encoder and corpus; the same holds at every budget from 4 to D on both encoders (all 24 cells), as it does for MiniLM (Table 6).

	SciFact		**FIQA**		**NFCorpus**	
Encoder	FP16	Binary	FP16	Binary	FP16	Binary
bge-small (384-d)	6.28	**71.24**	4.52	**33.21**	2.07	**13.38**
bge-large (1024-d)	6.22	**80.12**	4.88	**45.73**	2.22	**17.35**

within the compression regime. As a reproduction check, evaluating the fitted recall–retention curve f at the retention each budget affords recovers the observed storage-axis recalls: on SciFact, binary reaches 12.98/36.39/69.72% predicted versus 12.84/36.31/69.75% directly measured at 64/192/384 bits/document (Table 6), and FP16 is matched to within a few tenths of a point.

Table 6: SciFact Recall@10 at equal storage (random selection, 25 seeds). "Storage" is the absolute footprint for a 100M-document corpus; binary's 16× lower per-document cost retains proportionally more documents. NFCorpus and NQ follow the same pattern (Figure 1b).

Budget	Storage	**FP16**		**Binary**	
(bits/doc)	(100M corpus)	Docs (retention)	Recall@10	Docs (retention)	Recall@10
4	50 MB	3 (0.06%)	0.08%	53 (1.0%)	1.07%
64	800 MB	53 (1.0%)	1.12%	863 (16.7%)	12.84%
192	2.4 GB	161 (3.1%)	2.93%	2,591 (50.0%)	36.31%
384	4.8 GB	323 (6.2%)	5.85%	5,183 (100%)	69.75%

C Reranking Robustness

We retrieve the top-100 candidates from each budgeted index and rerank them with an off-the-shelf cross-encoder (`ms-marco-MiniLM-L-6-v2`), reporting post-rerank nDCG@10 with first-stage Recall@100 in parentheses, on SciFact and NFCorpus at the 4, 32, and 128 bit budgets (5 seeds, random selection). Table 7 reports the result: the cheaper PQ48×4 code beats binary at every budget after reranking, preserving the storage-equivalent ordering established under Recall@10.

Table 7: Effect of a downstream cross-encoder reranker (5 seeds, random selection). Each cell is post-rerank nDCG@10 with first-stage Recall@100 in parentheses (%). The cheaper PQ48×4 code beats binary at every budget after reranking.

Dataset	Budget	Binary	PQ48×4
SciFact	4	0.5 (0.5)	**0.8 (0.8)**
SciFact	32	6.7 (7.6)	**10.8 (12.5)**
SciFact	128	24.4 (28.9)	**47.8 (60.0)**
NFCorpus	4	2.2 (0.8)	**3.5 (1.6)**
NFCorpus	32	8.6 (4.7)	**14.5 (8.0)**
NFCorpus	128	18.8 (12.0)	**27.2 (20.7)**

D Heterogeneous Methods: Details

A PCA projection to d components stored at FP16 costs $16d$ bits/document, so at budget B it retains $\lfloor BN/(16d) \rfloor$ documents (random selection, d chosen per dataset on the validation split over $d \in \{16, 32, 64, 128, 256\}$). Placed on the common axis, PCA is dominated by product quantization at every budget on every corpus: spending the budget on coarse 4-bit codes across all 384 dimensions beats spending it on a few full-precision components, and the validation-selected dimension always lies at the small end of the sweep (16 or 32, never above), i.e. coverage is preferred to per-vector fidelity. PCA here is an untrained linear stand-in for a learned Matryoshka encoder (Kusupati et al., 2022), which might shift this balance. Two robustness checks: a data-oblivious Johnson–Lindenstrauss (Johnson & Lindenstrauss, 1984) Gaussian random projection does worse than PCA at nearly every point, so the verdict is not an artifact of a data-fitted reducer; and OPQ (Ge et al., 2014) tracks plain PQ48×4 to within ∼10% at every budget, so the conclusions do not depend on an unoptimized quantizer.

E Multi-Model Results

Table 8 gives the per-cell dense query-likelihood-vs-random comparison across the three encoders underlying §5.5.

Table 8: Recall@10 (%) at 4 bits/document across three encoders (25 seeds): dense query-likelihood (DQ) vs. random (Rand) selection, both at binary storage. * marks a DQ-vs-random difference significant at $p < 0.05$ after Holm–Bonferroni correction over the nine model×corpus comparisons (paired t-test). The selector's edge does not replicate across encoders: significant on different corpora for different models, within seed noise elsewhere.

	SciFact		**FIQA**		**NFCorpus**	
Encoder	DQ	Rand	DQ	Rand	DQ	Rand
MiniLM-L6 (384-d)	2.12*	1.07	0.49	0.79	0.89	0.60
bge-small (384-d)	1.87	1.12	0.77	0.77	0.58	0.59
bge-large (1024-d)	0.21	0.41	0.62*	0.33	0.45	0.38