EVERY BYTE MUST RANK: A RETRIEVAL-AWARE CODEC FOR FROZEN MULTIMODAL EMBEDDINGS

Anonymous authors
Paper under double-blind review

ABSTRACT

Multimodal retrieval systems increasingly store every candidate as a high-dimensional embedding, and the bill is paid per vector: one billion 2,048-dimensional FP32 embeddings occupy 8.2 TB before any index is built. This scale makes embedding storage a critical systems problem. Yet the tools used to compress these vectors were designed to reconstruct them. Product quantizers minimize reconstruction error even though retrieval depends on the resulting rankings, and fixed-width codes charge every symbol equally regardless of predictability. We introduce EmbedCodec, a codec trained for ranking rather than reconstruction. A lightweight adapter shared by queries and candidates is optimized through the hard product-quantizer assignments used at deployment, reshaping the frozen encoder's geometry for post-quantization retrieval. A task-conditioned Gaussian mixture models the resulting indices through codeword geometry; rANS then entropy-codes those indices, and rate is measured from serialized bytes rather than nominal bit width. Results on five retrieval tasks spanning text, cross-modal, and visual-document search with two frozen encoders show that EmbedCodec uses about 125 B/vector, a roughly $65\times$ storage reduction relative to FP32, while retaining 99.6–99.8% of the corresponding five-task average NDCG@10. At about 500 B/vector, it exceeds the FP32 average on both encoders. Every persisted stream is exactly decodable back to its input index tuples, and each of the ten encoder–task panels contains at least one Pareto-nondominated EmbedCodec operating point. Our findings suggest that retrieval-aware source and entropy coding can substantially reduce embedding storage without requiring faithful coordinate reconstruction.

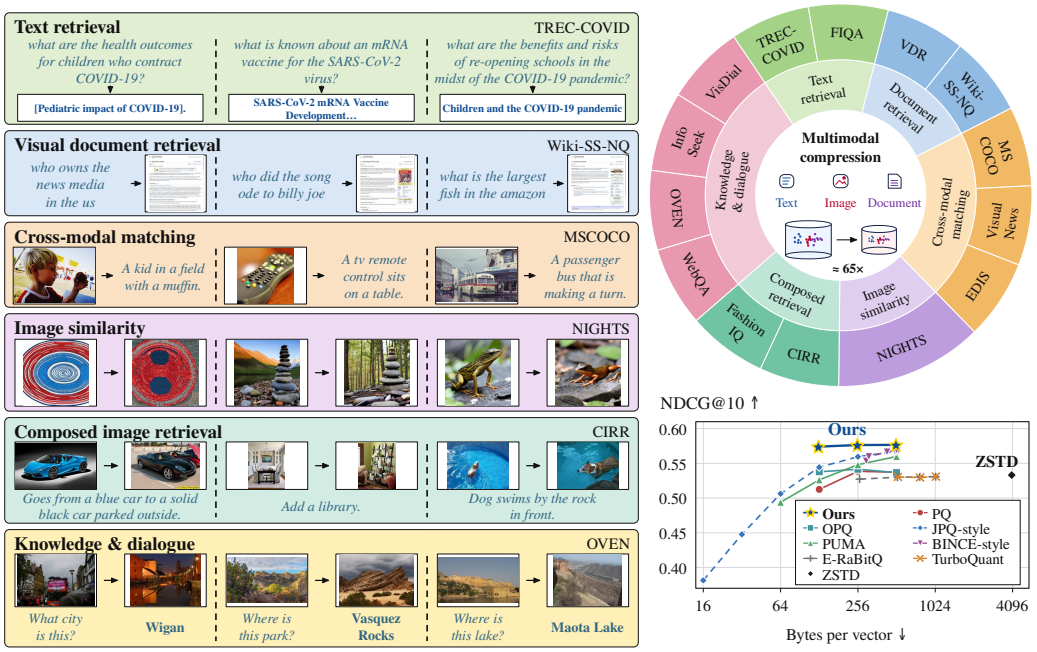
1 INTRODUCTION

Retrieval-augmented generation, enterprise knowledge bases, vector databases, and multimodal search increasingly materialize pretrained representations as large candidate collections (Wei et al., 2024; Jiang et al., 2025; Li et al., 2026). Public multimodal corpora already contain billions of items: LAION-5B reports 5.85 billion image–text pairs, and DataComp builds a candidate pool of 12.8 billion pairs (Schuhmann et al., 2022; Gadre et al., 2023). A single 2,048-dimensional FP32 embedding occupies 8,192 bytes, so one billion such vectors require 8.192 TB for the vectors alone. At this scale, embeddings are not disposable intermediate activations; they are a first-class storage object whose representation determines whether a retrieval system is economically deployable. For a collection of N candidates, every avoidable byte is paid N times before replication, versioning, or indexing is even considered. Embedding storage is therefore not a secondary systems detail: it determines which retrieval collections fit within a low-latency storage budget.

When every floating-point coordinate must be preserved exactly, a mature option is general-purpose lossless compression such as Zstandard (ZSTD). This choice is operationally attractive because decompression recovers the original vectors and requires no change to the embedding model. Its gain on dense floating-point arrays can nevertheless be modest. In a public ClickHouse measurement on a 10-million-row LAION embedding column, ZSTD reduces 28.59 GiB to 18.36 GiB, a compression ratio of about $1.56\times$.[1] The number is workload-specific, but it illustrates the underlying

[1] https://clickhouse.com/blog/vector-search-clickhouse-p2

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.



(a) Retrieval tasks and compression

(b) Multimodal retrieval rate–quality curve

Figure 1: **Multimodal retrieval tasks and compact embedding storage.** Panel (a) shows representative benchmark query–positive pairs from six application families and the shared compression bottleneck across text, image, and document embeddings. The central icon summarizes a representative $\approx 65\times$ storage reduction relative to FP32. Panel (b) illustrates the rate–quality trend on a pooled multi-domain retrieval benchmark spanning BEIR, MMEB, and VDR.

limitation: byte-level coding preserves every coordinate and therefore cannot remove redundancy in the embedding representation itself. Substantially lower storage rates require a controlled change of representation.

Existing lossy approaches make that change in several ways. Reduced-precision formats such as FP16 and BF16 halve coordinate storage, while binary or ternary codes reduce it further; aggressive settings, however, introduce clipping and rounding and can discard magnitude information (Ferdowsi et al., 2017; Gao & Long, 2024). Dimensionality reduction and nested representations remove coordinates altogether; they reduce both storage and comparison cost, but discarded subspaces cannot be recovered and their retrieval value can depend on the data and task (Kusupati et al., 2022; Dinu et al., 2025). Sparsification stores only selected coordinates, trading dense arithmetic for index and value storage while risking changes to the original similarity geometry (Ferdowsi et al., 2017; Attimonelli et al., 2026). Product quantization (PQ) replaces subvectors with codebook indices, and optimized PQ (OPQ) learns a rotation to reduce quantization error (Jégou et al., 2011; Ge et al., 2013); newer methods such as TurboQuant further improve geometric distortion at low precision (Zandieh et al., 2025). Finally, retrieval-aware methods optimize inner products, ranking losses, or constrained assignments rather than reconstruction alone (Guo et al., 2020; Dai et al., 2020; Zhan et al., 2021; Xiao et al., 2021; Zhan et al., 2022). These approaches expose a common limitation: they usually treat the stored code either as a geometric approximation used for retrieval or as a symbol sequence used for compression. In a deployed codec, however, the same hard code must serve both roles.

This dual role creates a single technical bottleneck at the quantizer interface. A reconstruction-trained quantizer need not preserve the comparisons used by retrieval, while a coder fitted only after quantization cannot change costly assignments. Moreover, centroid reconstruction generally moves candidates away from the unit sphere, so the scorer determines whether the resulting radial information is retained or discarded. The object to be learned is therefore not merely a compact embedding, but a discrete retrieval source whose assignments determine reconstruction, probability, and ranking.

We study this problem with frozen multimodal embedding encoders. Our goal is to reshape a fixed product partition without learning a larger decoder, assign geometrically meaningful lengths to its discrete cells, and preserve the geometry of the decoded candidates at retrieval time. This view separates what should be shared across tasks from what should remain task-specific: tasks share the residual adapter and quantizer, while their known candidate collections may induce different symbol statistics.

Figure 1 illustrates the resulting setting. Heterogeneous text, image, and document retrieval tasks share the same storage bottleneck, yet their candidate collections occupy different regions of a common rate–quality plane. The codec must therefore expose measured operating points rather than assuming that a wider nominal code is always more useful or equally compressible.

We introduce EMBEDCODEC, a retrieval-aware codec for frozen multimodal embeddings (Figure 2). For each encoder and rate, a residual adapter is shared by queries, candidates, and all tasks. Candidates then pass through normalization and hard assignment to a fixed OPQ codebook, whereas queries remain continuous in the same adapted space. The synthesis map remains centroid lookup followed by inverse rotation. Training the adapter therefore learns the pullback of a fixed product partition: it changes which embeddings share a stored code without moving the codewords or adding a learned decoder.

The selected cells also define the discrete source to be stored. We evaluate a task-conditioned GMM on centroid vectors, rather than on arbitrary integer labels, and normalize these densities over each finite codebook. The resulting code lengths are invariant to a consistent relabeling of codewords. Through soft assignment, their gradient is a covariance between centroid location and code length, so retrieval supervision and rate act on the same assignment boundaries. Task identity specializes only the probability table; it does not route the adapter, quantizer, or placement, and it introduces no per-vector routing bits.

At retrieval time, EMBEDCODEC preserves the geometry produced by the fixed synthesis map. Decoded candidates are not projected back onto the unit sphere; their reconstructed norms remain in the negative squared ℓ_2 score against continuous adapted queries. After the hard index tuple is fixed, block rANS recovers it exactly (Duda, 2013). Entropy coding can then change the stored length but not the decoded candidate. We measure this length from the serialized `main`-plus-`hyper` bitstream rather than from nominal index width or model likelihood.

We evaluate this design on FIQA, COCO text-to-image and image-to-text retrieval, VDR-en, and WebQA with Qwen3-VL-Embedding-2B and Seed1.6-Embedding. At the lowest measured rate, EmbedCodec reduces the 8,192-B FP32 representation by $65.6\times$ on Qwen3-VL-Embedding-2B and $65.0\times$ on Seed1.6-Embedding, while retaining 99.6% and 99.8% of their respective FP32 five-task average NDCG@10. On the pooled multi-domain benchmark, the Qwen configuration occupies 129.837 B/vector versus 4034.179 B/vector for lossless FP32+ZSTD, a $31.071\times$ reduction, while NDCG@10 increases from 0.533 to 0.573. Of 30 measured operating points, 22 lie on the storage–quality Pareto frontier: no evaluated alternative attains equal or higher NDCG@10 with equal or fewer bytes. These results expose a task-dependent quantized source: increasing nominal width does not uniformly improve retrieval, and sharing the adapted geometry does not make symbol statistics or decoded scoring behavior task-neutral.

Our contributions are: **(1) Fixed-partition retrieval adaptation.** We formulate a pre-OPQ residual adapter as learning the pullback of a fixed OPQ partition. Hard reconstructed candidates receive retrieval supervision, while synthesis remains fixed without adding a learned decoder. **(2) Geometric source shaping.** We define a task-conditioned, label-permutation-invariant GMM over codeword vectors and derive the covariance form of its rate gradient. This couples retrieval utility and code length at the hard-assignment interface. **(3) A characterization of task-dependent quantized retrieval.** We show that a shared residual adapter does not produce a task-neutral source: different retrieval directions exhibit sharply different finite-stream compressibility, and the norms induced by centroid reconstruction remain relevant to ranking. This characterization motivates task-specific entropy tables together with a shared representation and a norm-retaining deployment scorer.

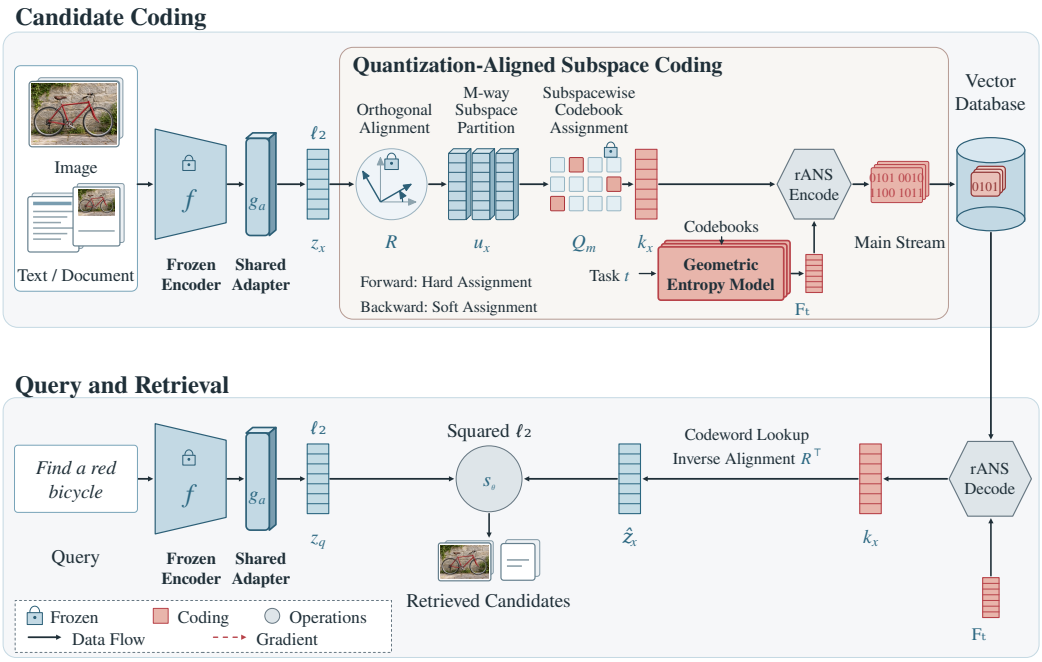


Figure 2: **Candidate coding and query-time retrieval.** A shared adapter learns a normalized retrieval geometry at the input to a fixed subspace quantizer. Orthogonal alignment, subspace partition, and subspacewise nearest-codeword assignment produce discrete candidate indices. A task-conditioned geometric GMM supplies their probabilities for rANS coding. Training uses normalized-prefix cosine losses, whereas EMBEDCODEC's deployment ranks the decoded candidates by negative squared ℓ_2 distance without renormalizing their reconstructed vectors. Dashed arrows show training signals to the adapter and entropy model; the source embedding encoder, alignment, and codebooks stay fixed.

2 RELATED WORK

Compact retrieval representations. Multimodal embedding models map heterogeneous query–candidate pairs into a shared space (Wei et al., 2024; Jiang et al., 2025; Li et al., 2026). Matryoshka representations, low-precision codes, and learned sparsity reduce dimensions, precision, or support (Kusupati et al., 2022; Dinu et al., 2025; Ferdowsi et al., 2017; Attimonelli et al., 2026). Our setting instead keeps the backbone frozen and trains through the hard representation used after decoding.

Retrieval-aware quantization. PQ and OPQ encode subvectors with codebook indices (Jégou et al., 2011; Ge et al., 2013); RaBitQ and TurboQuant improve low-bit geometric fidelity (Gao & Long, 2024; Zandieh et al., 2025). Retrieval-oriented methods optimize inner products, ranking losses, or assignments (Guo et al., 2020; Dai et al., 2020; Zhan et al., 2021; Xiao et al., 2021; Zhan et al., 2022). EMBEDCODEC likewise targets retrieval, but learns a pre-OPQ geometry whose assignments determine both ranking and rate.

Entropy coding. DeltaPQ and SegPQ compress structured PQ indices losslessly (Wang & Deng, 2020; Liu et al., 2025), while learned compression combines quantized latents with probability models and entropy coders (Ballé et al., 2018; 2021; Cheng et al., 2020; He et al., 2022). EMBEDCODEC connects these lines at the quantizer interface: a geometric GMM assigns task-conditioned probabilities to fixed codewords, and retrieval is evaluated after exact stream decoding. Extended discussion appears in the supplementary material.

3 METHOD

Learning at the quantizer interface. A fixed subspace quantizer assigns each candidate to a tuple of codewords. The geometry presented to this quantizer determines both the reconstructed candidate and the frequency of its discrete indices. We therefore learn the input geometry using two signals: retrieval supervision on reconstructed candidates and a rate for the selected indices. Figure 2 shows this shared learning path. For a frozen multimodal embedding $\mathbf{v}_x = f(x) \in \mathbb{R}^D$, an adapter with parameters θ produces an index tuple $\mathbf{k}_x$, to which an entropy model with parameters ψ assigns a code length to. The task-constrained formulation is

$$\min_{\theta,\psi} \mathbb{E}[\mathcal{R}_\psi(\mathbf{k}_x \mid t)] \quad \text{s.t.} \quad \mathbb{E}[\mathcal{L}_{\text{ret}}(q, \mathcal{C}; \theta)] \le \epsilon, \tag{1}$$

where t identifies the retrieval task, $\mathcal{C}$ is a candidate set, and $\mathcal{R}_\psi$ is the model code length. As in task-oriented learned compression, we optimize a rate–task Lagrangian as a tractable surrogate and measure the resulting serialized bytes separately. Retrieval supervision encourages useful rankings after quantization; it does not guarantee invariance of individual decisions.

Retrieval-aware geometry adaptation. Queries and candidates share a residual transform followed by normalization:

$$\mathbf{r}_x = g_a(\mathbf{v}_x; \theta) = \mathbf{v}_x + \mathbf{W}_2 \,\mathrm{ReLU}(\mathbf{W}_1 \mathbf{v}_x + \mathbf{b}_1) + \mathbf{b}_2, \qquad \mathbf{z}_x = \frac{\mathbf{r}_x}{\|\mathbf{r}_x\|_2}. \tag{2}$$

Both $\mathbf{r}_x$ and $\mathbf{z}_x$ depend on θ; we suppress this argument on intermediate vectors when it is unambiguous. Here $\mathbf{W}_1 \in \mathbb{R}^{H \times D}$, $\mathbf{b}_1 \in \mathbb{R}^H$, $\mathbf{W}_2 \in \mathbb{R}^{D \times H}$, and $\mathbf{b}_2 \in \mathbb{R}^D$, where H is the adapter hidden dimension. The output projection is initialized to zero, so g_a starts as the identity map. Placing this trainable adapter before assignment lets retrieval and rate supervision change which codewords represent a candidate; applying the same adapter to queries keeps the two sides in a common geometry. We use this pre-quantization placement throughout. Within each encoder–rate configuration, one adapter is shared across all tasks; neither the adapter nor its placement is selected by the task.

Quantization-aligned subspace coding. The normalized candidate passes through three successive operations. An orthogonal matrix $\mathbf{R}$ aligns its coordinates, the aligned vector is partitioned into M equal subspaces, and a codebook assigns one index to each subvector:

$$\begin{aligned} \mathbf{u}_x &= \mathbf{R}\mathbf{z}_x = [\mathbf{u}_{x,1}; \ldots; \mathbf{u}_{x,M}], \qquad \mathbf{R}^\top \mathbf{R} = \mathbf{I}_D, \\ k_{x,m} &= Q_m(\mathbf{u}_{x,m}) = \arg\min_{j \in \{1,\ldots,K\}} \|\mathbf{u}_{x,m} - \mathbf{c}_{m,j}\|_2^2 . \end{aligned} \tag{3}$$

Here $\mathbf{c}_{m,j} \in \mathbb{R}^{D/M}$ is codeword j in subspace m. The tuple $\mathbf{k}_x = (k_{x,1}, \ldots, k_{x,M})$ has a nominal fixed-width cost of $M\lceil \log_2 K \rceil$ bits. We instantiate the alignment and codebooks with OPQ (Ge et al., 2013), fitted once to frozen training-candidate embeddings before adapter optimization and then held fixed. The three operations together define the product quantizer through which both retrieval and rate signals reach g_a.

Decoding looks up the selected codewords, concatenates them, and reverses the alignment. The continuous query $\mathbf{z}_q(\theta)$ is scored against this reconstruction:

$$\hat{\mathbf{z}}_x(\theta) = \mathcal{S}_{\text{OPQ}}(\mathbf{k}_x), \qquad s_\theta(q, \mathbf{k}_x) = -\|\mathbf{z}_q(\theta) - \hat{\mathbf{z}}_x(\theta)\|_2^2 . \tag{4}$$

The unit-normalized query bypasses codebook assignment, while the reconstruction retains its decoded norm and is not normalized again before scoring. Since $\|\mathbf{z}_q(\theta)\|_2 = 1$, the score decomposes as

$$s_\theta(q, \mathbf{k}_x) = -1 - \|\hat{\mathbf{z}}_x(\theta)\|_2^2 + 2\mathbf{z}_q(\theta)^\top \hat{\mathbf{z}}_x(\theta). \tag{5}$$

The retained norm therefore contributes to ranking instead of being removed by a second normalization. Equation (4) defines the deployment scorer; reconstruction requires only codebook lookup and the inverse alignment.

Gradients through discrete assignment. Hard assignment makes the input-to-codeword map piecewise constant. During training, we retain the selected codeword in the forward pass and differentiate through a soft assignment:

$$\begin{aligned}\pi_{x,m,j} &= \frac{\exp(-\left\|\mathbf{u}_{x,m}-\mathbf{c}_{m,j}\right\|_2^2/\tau_{\mathrm{a}})}{\sum_{\ell=1}^{K}\exp(-\left\|\mathbf{u}_{x,m}-\mathbf{c}_{m,\ell}\right\|_2^2/\tau_{\mathrm{a}})},\\ \tilde{\mathbf{c}}_{x,m} &= \sum_{j=1}^{K}\pi_{x,m,j}\mathbf{c}_{m,j}, \qquad \mathbf{c}_{x,m}^{\mathrm{ST}} = \tilde{\mathbf{c}}_{x,m}+\mathrm{sg}(\mathbf{c}_{m,k_{x,m}}-\tilde{\mathbf{c}}_{x,m}).\end{aligned} \tag{6}$$

Here sg stops gradients and $\tau_{\mathrm{a}} > 0$ is the assignment temperature. This is a straight-through estimator (STE): substituting $\mathbf{c}^{\mathrm{ST}}$ into reconstruction exposes the hard-quantized candidate to the retrieval loss, while its backward path updates the adapter through distances to the fixed codewords.

Task-conditioned geometric entropy model. The adapter also controls the distribution of the selected indices. To supply a coding cost on the same codebook geometry, each task t and subspace m has a K_{GMM}-component diagonal Gaussian mixture:

$$\begin{aligned}h_{\psi,t,m}(\mathbf{c}) &= \sum_{\ell=1}^{K_{\mathrm{GMM}}} w_{t,m,\ell}\mathcal{N}\big(\mathbf{c};\boldsymbol{\mu}_{t,m,\ell},\mathrm{diag}(\boldsymbol{\sigma}_{t,m,\ell}^2)\big),\\ p_\psi(k_m=j\mid t) &= \frac{h_{\psi,t,m}(\mathbf{c}_{m,j})}{\sum_{a=1}^{K}h_{\psi,t,m}(\mathbf{c}_{m,a})}, \qquad p_\psi(\mathbf{k}_x\mid t)=\prod_{m=1}^{M}p_\psi(k_{x,m}\mid t).\end{aligned} \tag{7}$$

Here $w_{t,m,\ell}$ is a scalar mixture weight, while $\boldsymbol{\mu}_{t,m,\ell}, \boldsymbol{\sigma}_{t,m,\ell} \in \mathbb{R}^{D/M}$ are the component mean and coordinatewise standard deviation; the weights sum to one over ℓ. The Gaussian density is evaluated on the vector $\mathbf{c}_{m,j} \in \mathbb{R}^{D/M}$, not on the integer label j; labels serve only as codebook addresses. A consistent permutation of labels and centroid rows therefore leaves the model unchanged. Normalizing these geometric densities over the finite codebook gives the discrete symbol probabilities required by the entropy coder. The task is known from the candidate collection and selects the corresponding probability tables at both encoder and decoder. Task identity affects only this entropy-table lookup: it does not route examples to different adapters, quantizers, or adapter placements, and it adds no per-vector routing bits. These tables are static at deployment; the adapter and quantizer topology are shared across tasks.

We use $\mathcal{R}_\psi(\mathbf{k}\mid t)$ throughout for model code length; tildes and the superscript "ST" identify only its training surrogates. The model rate in bits per candidate and its adapter-gradient surrogate are

$$\begin{aligned}\mathcal{R}_\psi(\mathbf{k}_x\mid t) &= -\sum_{m=1}^{M}\log_2 p_\psi(k_{x,m}\mid t),\\ \widetilde{\mathcal{R}}_\psi(\mathbf{v}_x\mid t) &= -\sum_{m=1}^{M}\sum_{j=1}^{K}\pi_{x,m,j}\,\mathrm{sg}\big(\log_2 p_\psi(k_m=j\mid t)\big),\\ \mathcal{R}_\psi^{\mathrm{ST}}(\mathbf{v}_x\mid t) &= \mathcal{R}_\psi(\mathbf{k}_x\mid t)+\widetilde{\mathcal{R}}_\psi(\mathbf{v}_x\mid t)-\mathrm{sg}\Big(\widetilde{\mathcal{R}}_\psi(\mathbf{v}_x\mid t)\Big).\end{aligned} \tag{8}$$

The forward value is the code length of the hard indices. Its gradient fits the GMM to those indices and updates g_a through the soft assignment probabilities. Together with retrieval supervision, this gives the adapter a coding preference among nearby assignments while penalizing changes that harm the training rankings.

The adapter-side effect of the entropy model can be stated explicitly. For one subspace, define the task-dependent code length $\ell_{m,j}^{(t)} = -\log_2 p_\psi(k_m = j \mid t)$ and its soft local average $\bar{\ell}_{x,m}^{(t)} = \sum_j \pi_{x,m,j}\ell_{m,j}^{(t)}$. With the code lengths stopped as in Equation (8), differentiation through the soft assignment gives

$$\nabla_{\mathbf{u}_{x,m}}\widetilde{\mathcal{R}}_{\psi,m}(\mathbf{v}_x\mid t) = \frac{2}{\tau_{\mathrm{a}}}\sum_{j=1}^{K}\pi_{x,m,j}\left(\ell_{m,j}^{(t)}-\bar{\ell}_{x,m}^{(t)}\right)\mathbf{c}_{m,j} = \frac{2}{\tau_{\mathrm{a}}}\,\mathrm{Cov}_{j\sim\pi_{x,m}}\left(\ell_{m,j}^{(t)},\mathbf{c}_{m,j}\right). \tag{9}$$

The covariance is taken under the soft categorical weights $\pi_{x,m}$. The code lengths define a scalar field over the centroids. If all symbols receiving non-negligible soft-assignment mass have the same code length, its gradient is zero. Otherwise, the covariance between code length and centroid location determines the rate gradient, while the retrieval loss discourages assignment changes that damage ranking.

Joint training. Let $\mathbf{A} = [\mathbf{a}_1^\top; \ldots; \mathbf{a}_B^\top]$ and $\mathbf{B} = [\mathbf{b}_1^\top; \ldots; \mathbf{b}_B^\top]$ be batches of query and candidate vectors. We use a query-to-candidate, multi-positive contrastive loss over independently normalized prefixes:

$$\mathcal{L}_{\mathrm{con}}(\mathbf{A}, \mathbf{B}) = -\frac{1}{B|\mathcal{D}|} \sum_{i=1}^{B} \sum_{d \in \mathcal{D}} \log \frac{\sum_{j \in \mathcal{P}_i} \exp(\cos(\mathbf{a}_i^{:d}, \mathbf{b}_j^{:d})/\tau_{\mathrm{c}})}{\sum_{j=1}^{B} \exp(\cos(\mathbf{a}_i^{:d}, \mathbf{b}_j^{:d})/\tau_{\mathrm{c}})}. \tag{10}$$

Let $\mathcal{Y}(q_i)$ be the task-defined training-positive candidate set for query q_i. Then $\mathcal{P}_i = \{j : x_j \in \mathcal{Y}(q_i)\}$ contains every occurrence of those positives in the current minibatch; it does not import relevance labels from the evaluation corpus. This multi-positive construction avoids treating another task-defined positive in the batch as a negative. The scalar $\tau_{\mathrm{c}} > 0$ is the contrastive temperature, and $\mathcal{D}$ is the set of supervised prefix dimensions. The cold loss $\mathcal{L}_{\mathrm{cold}} = \mathcal{L}_{\mathrm{con}}(\{\mathbf{r}_{q_i}\}_{i=1}^B, \{\hat{\mathbf{z}}_{x_j}(\theta)\}_{j=1}^B)$ supervises reconstructed candidates. The auxiliary sign-prefix loss $\mathcal{L}_{\mathrm{hot}} = \mathcal{L}_{\mathrm{con}}(\{\mathrm{sign}(\mathbf{r}_{q_i})\}_{i=1}^B, \{\mathrm{sign}(\mathbf{r}_{x_j})\}_{j=1}^B)$ uses the unquantized adapter outputs, with $\mathrm{sign}(0) = +1$ and straight-through gradients for sign. Both losses use prefix-normalized cosines as training surrogates; they do not define the deployment scorer, which uses the full-vector distance in Equation (4). A two-sided cosine identity penalty controls drift from the frozen embeddings:

$$\mathcal{L}_{\mathrm{id}} = \frac{1}{B} \sum_i \left[1 - \cos(\mathbf{r}_{q_i}, \mathbf{v}_{q_i}) + 1 - \cos(\mathbf{r}_{x_i}, \mathbf{v}_{x_i})\right]. \tag{11}$$

Thus $\mathcal{L}_{\mathrm{ret}} = \mathcal{L}_{\mathrm{cold}} + \mathcal{L}_{\mathrm{hot}} + \mathcal{L}_{\mathrm{id}}$, consistent with Equation (1). The adapter and GMM are jointly optimized with

$$\mathcal{L} = \mathcal{L}_{\mathrm{ret}} + \lambda B^{-1} \sum_i \mathcal{R}_\psi^{\mathrm{ST}}(\mathbf{v}_{x_i} \mid t_i), \qquad \lambda > 0. \tag{12}$$

During joint training, g_a and the GMM parameters are trainable, while the source embedding encoder, rotation, and codebooks remain fixed. After training, the adapter and the resulting probability tables are frozen for encoding and retrieval.

Entropy coding. The GMM probabilities are quantized into 16-bit cumulative-distribution function (CDF) tables for rANS. A K-symbol codebook plus one reserved escape symbol defines $K+1$ coding intervals and therefore requires $K+2$ cumulative boundaries, including the initial zero and terminal count. Let $\mathbf{F}_t \in \mathbb{N}^{M \times (K+2)}$ collect the task-t CDFs, and let $\mathbf{K}_{t,b} \in \{1, \ldots, K\}^{N_{t,b} \times M}$ stack the index tuples in block b, where $N_{t,b} \le B_c$ and $B_c = 256$ in our experiments. Each block uses rANS unless entropy coding would exceed its uncompressed fixed-width representation, in which case the encoder stores the raw indices. Writing $\boldsymbol{\eta}_{t,b}$ for the decoder-side block metadata (coding mode, byte offset, and row count), the implemented block interface satisfies

$$\begin{aligned} (\mathtt{main}_{t,b}, \boldsymbol{\eta}_{t,b}) &= \mathrm{Enc}(\mathbf{K}_{t,b}; \mathbf{F}_t), \\ \mathrm{Dec}(\mathtt{main}_{t,b}; \mathbf{F}_t, \boldsymbol{\eta}_{t,b}) &= \mathbf{K}_{t,b}. \end{aligned} \tag{13}$$

Thus every recovered tuple $\tilde{\mathbf{k}}_x$ equals its input $\mathbf{k}_x$. Combining this identity with the fixed OPQ synthesis map yields the separation

$$\hat{\mathbf{z}}_x^{\mathrm{stored}}(\theta) = \mathcal{S}_{\mathrm{OPQ}}(\tilde{\mathbf{k}}_x) = \mathcal{S}_{\mathrm{OPQ}}(\mathbf{k}_x) = \mathcal{S}_{\mathrm{OPQ}}(Q(\mathbf{R}n(g_a(\mathbf{v}_x; \theta)))). \tag{14}$$

Hence the analysis–synthesis path determines the reconstructed candidate and its retrieval distortion, whereas the entropy layer changes only the number of stored bits. Retrieval and rate interact before hard assignment through g_a and Equation (9). After the product quantizer produces $\mathbf{k}_x$, entropy coding cannot alter the reconstruction. Exact index recovery checks this property, while serialized bytes measure storage. Entropy coding therefore changes the stored length but introduces no distortion beyond the fixed synthesis reconstruction. Section 4 specifies the byte-accounting protocol used in the experiments.

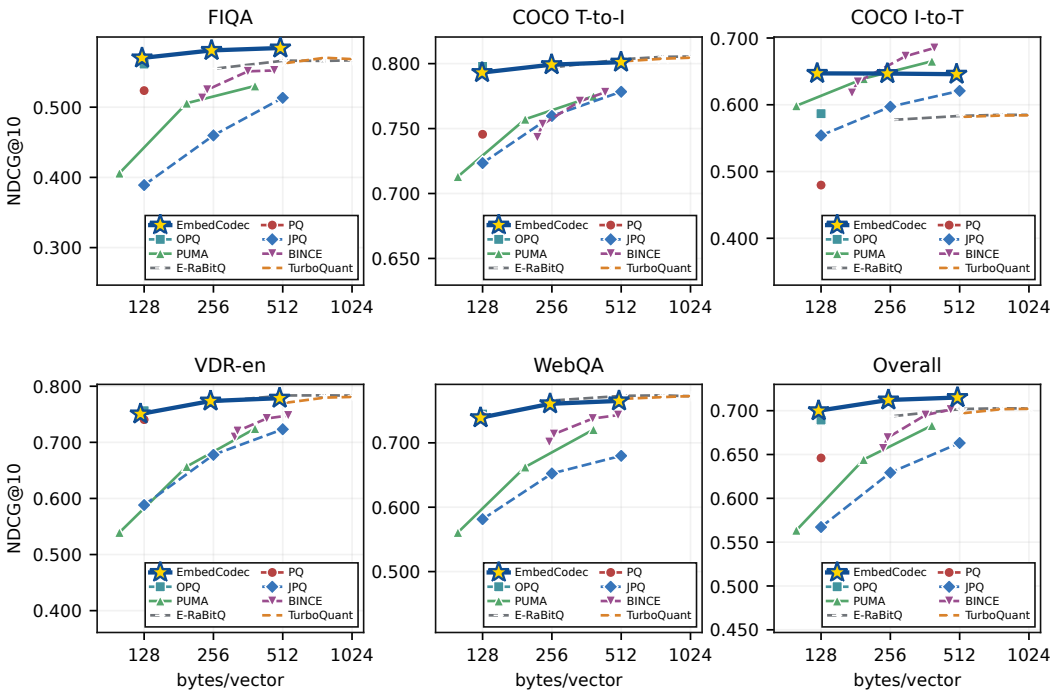


Figure 3: **Full-corpus storage–quality curves for Qwen3-VL-Embedding-2B.** The first five panels show the individual retrieval tasks, and the final panel shows their unweighted average. Each line connects measurements only; yellow stars denote EMBEDCODEC. Full-range plots are supplementary. JPQ uses native asymmetric-inner-product ADC and codes-payload primary rate; EMBEDCODEC uses decoded-vector negative squared ℓ_2 and main-plus-hyper stream rate. The comparison is descriptive, not same-scorer or complete-system-storage matched, and carries no significance claim.

4 EXPERIMENTS

Protocol. We evaluate FIQA, COCO text-to-image, COCO image-to-text, VDR-en, and WebQA using their complete candidate pools and report NDCG@10 per task and as an unweighted average. We compress 2,048-dimensional embeddings from frozen Qwen3-VL-Embedding-2B (Li et al., 2026) and Seed1.6-Embedding. For each encoder, $M \in \{128, 256, 512\}$ and $K = 256$ define the three measured rates. Each encoder–M configuration uses one pre-OPQ adapter and one fixed OPQ state across all tasks; task identity selects only its GMM/CDF table. EMBEDCODEC ranks decoded candidates by negative squared ℓ_2 without renormalization or FP32 reranking. Baselines retain their native scorers and declared primary rates, so the plots compare measured operating points rather than imposing a common post-hoc scorer. Training schedules, original query counts, target and realized task sampling, uncertainty procedures, and all accounting exclusions are reported in the supplementary material.

Storage–quality trade-off. At the lowest-rate setting, EMBEDCODEC uses 124.905 and 126.086 B/vector for Qwen3-VL-Embedding-2B and Seed1.6-Embedding, reductions of $65.6\times$ and $65.0\times$ from the 8,192-B FP32 representation. These points retain 99.6% and 99.8% of the corresponding FP32 five-task average NDCG@10. At the highest measured rate, 500.948 and 501.666 B/vector, the five-task averages reach 0.715 and 0.788 and exceed the corresponding FP32 averages. Every one of the ten encoder–task panels contains at least one Pareto-nondominated EMBEDCODEC operating point. The task curves are not uniformly monotone: Qwen COCO image-to-text and Seed FIQA peak at the low-rate setting, so all measured points remain visible rather than being summarized by nominal width.

Ablations and coding checks. The fixed pre-OPQ adapter improves over no adapter in all six encoder–width cells by 0.004–0.015 average NDCG@10. Negative squared ℓ_2 exceeds cosine in 24 of 30 task–width cells and five of six average cells. GMM-rANS reduces mean fixed-width

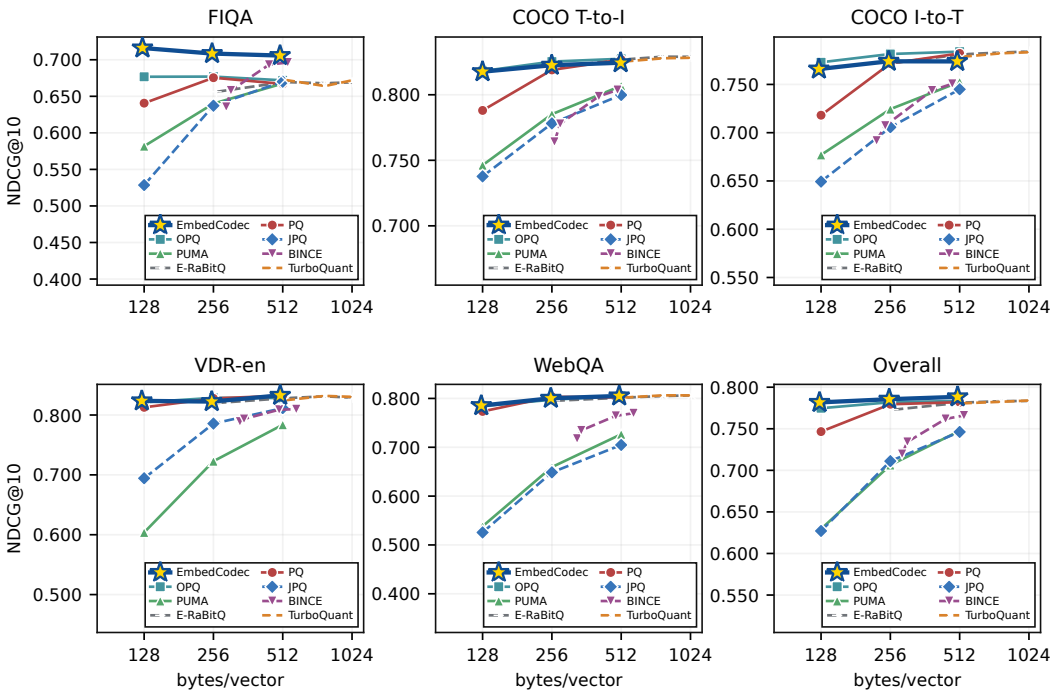


Figure 4: **Full-corpus storage–quality curves for Seed1.6-Embedding.** The first five panels show the individual retrieval tasks, and the final panel shows their unweighted average. Each line connects measurements only; yellow stars denote EMBEDCODEC. Full-range plots are supplementary. JPQ uses native asymmetric-inner-product ADC and codes-payload primary rate; EMBEDCODEC uses decoded-vector negative squared ℓ_2 and main-plus-hyper stream rate. The comparison is descriptive, not same-scorer or complete-system-storage matched, and carries no significance claim.

index cost by 1.167–2.418% across the six encoder–width settings. Across all 30 streams, 11,763 blocks use GMM-rANS and 15 use raw fallback; every persisted stream exactly recovers its input index matrix, yielding a post-decoding NDCG@10 change of 0.000. The supplementary material reports the complete placement and scorer ablations, task-conditioned entropy-table controls, pooled multi-domain benchmark, failure cases, and full-range curves.

5 CONCLUSION

We asked whether a frozen multimodal embedding must be stored as floating-point coordinates to remain useful for retrieval. The results show that it need not, provided the code is learned for the ranking it serves. EMBEDCODEC trains a shared query–candidate adapter through the hard product quantizer used at deployment, assigns code lengths with a task-conditioned geometric GMM over codewords, and stores the resulting indices in exactly decodable rANS streams. Measured in serialized bytes, the codec uses roughly 125 B/vector, a roughly $65\times$ storage reduction relative to FP32, while retaining 99.6–99.8% of the corresponding five-task average NDCG@10; at roughly one-sixteenth the storage, it exceeds the FP32 average for both evaluated encoders. This gain has a clear boundary: when codewords are equally costly, the rate gradient vanishes, so compression depends on task-specific symbol redundancy rather than a universal guarantee. Richer entropy models and conditional codes could extend the same quantizer interface, while serialized-byte reporting should remain standard whenever embeddings are stored at scale.

REFERENCES

Matteo Attimonelli, Alessandro De Bellis, Franco Maria Nardini, Claudio Pomo, Cosimo Rulli, Rossano Venturini, and Tommaso Di Noia. PUMA: Post-hoc sparsification of universal multimodal embeddings for efficient retrieval, 2026. URL https://arxiv.org/abs/2608.25780. arXiv preprint.

Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. In *International Conference on Learning Representations*, 2018. URL https://arxiv.org/abs/1802.01436.

Johannes Ballé, Philip A. Chou, David Minnen, Saurabh Singh, Nick Johnston, Eirikur Agustsson, Sung Jin Hwang, and George Toderici. Nonlinear transform coding. *IEEE Journal of Selected Topics in Signal Processing*, 15, 2021. URL https://arxiv.org/abs/2007.03034.

Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. Learned image compression with discretized gaussian mixture likelihoods and attention modules. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. URL https://arxiv.org/abs/2001.01568.

Xinyan Dai, Xiao Yan, Kelvin K. W. Ng, Jie Liu, and James Cheng. Norm-explicit quantization: Improving vector quantization for maximum inner product search. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(01):51–58, 2020. doi: 10.1609/aaai.v34i01.5333. URL https://ojs.aaai.org/index.php/AAAI/article/view/5333.

Georgiana Dinu, Corey Barrett, Yi Xiang, Miguel Romero Calvo, Anna Currey, and Xing Niu. Effective post-training embedding compression via temperature control in contrastive training. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=szRmEM8Kx5.

Jarek Duda. Asymmetric numeral systems: Entropy coding combining speed of Huffman coding with compression rate of arithmetic coding, 2013. URL https://arxiv.org/abs/1311.2540.

Sohrab Ferdowsi, Slava Voloshynovskiy, Dimche Kostadinov, and Taras Holotyak. Sparse ternary codes for similarity search have higher coding gain than dense binary codes. In *2017 IEEE International Symposium on Information Theory*, pp. 2653–2657, 2017. doi: 10.1109/ISIT.2017.8007010. URL https://arxiv.org/abs/1701.07675.

Samir Yitzhak Gadre, Gabriel Ilharco, Alex Fang, Jonathan Hayase, Georgios Smyrnis, Thao Nguyen, Ryan Marten, Mitchell Wortsman, Dhruba Ghosh, Jieyu Zhang, Eyal Orgad, Rahim Entezari, Giannis Daras, Sarah Pratt, Vivek Ramanujan, Yonatan Bitton, Kalyani Marathe, Stephen Mussmann, Richard Vencu, Mehdi Cherti, Ranjay Krishna, Pang Wei Koh, Olga Saukh, Alexander J. Ratner, Shuran Song, Hannaneh Hajishirzi, Ali Farhadi, Romain Beaumont, Sewoong Oh, Alex Dimakis, Jenia Jitsev, Yair Carmon, Vaishaal Shankar, and Ludwig Schmidt. DataComp: In search of the next generation of multimodal datasets. In *Advances in Neural Information Processing Systems*, volume 36, pp. 27092–27112, 2023. doi: 10.52202/075280-1179. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/56332d41d55ad7ad8024aac625881be7-Abstract-Datasets_and_Benchmarks.html.

Jianyang Gao and Cheng Long. RaBitQ: Quantizing high-dimensional vectors with a theoretical error bound for approximate nearest neighbor search, 2024. URL https://arxiv.org/abs/2405.12497. Accepted at SIGMOD 2024.

Tiezheng Ge, Kaiming He, Qifa Ke, and Jian Sun. Optimized product quantization for approximate nearest neighbor search. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2946–2953, 2013. URL https://openaccess.thecvf.com/content_cvpr_2013/html/Ge_Optimized_Product_Quantization_2013_CVPR_paper.html.

Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar. Accelerating large-scale inference with anisotropic vector quantization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 3887–3896. PMLR, 2020. URL https://proceedings.mlr.press/v119/guo20h.html.

Dailan He, Ziming Yang, Weikun Peng, Rui Ma, Hongwei Qin, and Yan Wang. ELIC: Efficient learned image compression with unevenly grouped space-channel contextual adaptive coding. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. URL https://arxiv.org/abs/2203.10886.

Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011. doi: 10.1109/TPAMI.2010.57.

Ziyan Jiang, Rui Meng, Xinyi Yang, Semih Yavuz, Yingbo Zhou, and Wenhu Chen. VLM2Vec: Training vision-language models for massive multimodal embedding tasks. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.05160.

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, and Ali Farhadi. Matryoshka representation learning. In *Advances in Neural Information Processing Systems*, 2022. URL https://arxiv.org/abs/2205.13147.

Mingxin Li, Yanzhao Zhang, Dingkun Long, Keqin Chen, Sibo Song, Shuai Bai, Zhibo Yang, Pengjun Xie, An Yang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Qwen3-VL-Embedding and Qwen3-VL-Reranker: A unified framework for state-of-the-art multimodal retrieval and ranking. *arXiv preprint*, 2026. URL https://arxiv.org/abs/2601.04720.

Qiyu Liu, Yanlin Qi, Siyuan Han, Jingshu Peng, Jin Li, and Lei Chen. Not small enough? SegPQ: A learned approach to compress product quantization codebooks. *Proceedings of the VLDB Endowment*, 18(11):3730–3743, 2025. doi: 10.14778/3749646.3749650. URL https://www.vldb.org/pvldb/vol18/p3730-liu.pdf.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5b: An open large-scale dataset for training next generation image-text models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 25278–25294, 2022. doi: 10.52202/068431-1833. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/a1859debfb3b59d094f3504d5ebb6c25-Abstract-Datasets_and_Benchmarks.html.

Runhui Wang and Dong Deng. DeltaPQ: Lossless product quantization code compression for high dimensional similarity search. *Proceedings of the VLDB Endowment*, 13(13):3603–3616, 2020. doi: 10.14778/3424573.3424580. URL https://www.vldb.org/pvldb/vol13/p3603-wang.pdf.

Cong Wei, Yang Chen, Haonan Chen, Hexiang Hu, Ge Zhang, Jie Fu, Alan Ritter, and Wenhu Chen. UniIR: Training and benchmarking universal multimodal information retrievers. In *European Conference on Computer Vision*, 2024. URL https://arxiv.org/abs/2311.17136.

Shitao Xiao, Zheng Liu, Yingxia Shao, Defu Lian, and Xing Xie. Matching-oriented embedding quantization for ad-hoc retrieval. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 8119–8129, 2021. doi: 10.18653/v1/2021.emnlp-main.640. URL https://aclanthology.org/2021.emnlp-main.640/.

Amir Zandieh, Majid Daliri, Majid Hadian, and Vahab Mirrokni. TurboQuant: Online vector quantization with near-optimal distortion rate, 2025. URL https://arxiv.org/abs/2504.19874.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. Jointly optimizing query encoder and product quantization to improve retrieval performance. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*, 2021. doi: 10.1145/3459637.3482358. URL https://arxiv.org/abs/2108.00644.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. Learning discrete representations via constrained clustering for effective and efficient dense retrieval. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pp. 1328–1336, 2022. doi: 10.1145/3488560.3498443. URL https://arxiv.org/abs/2110.05789.