Embedding Compression via Spherical Coordinates

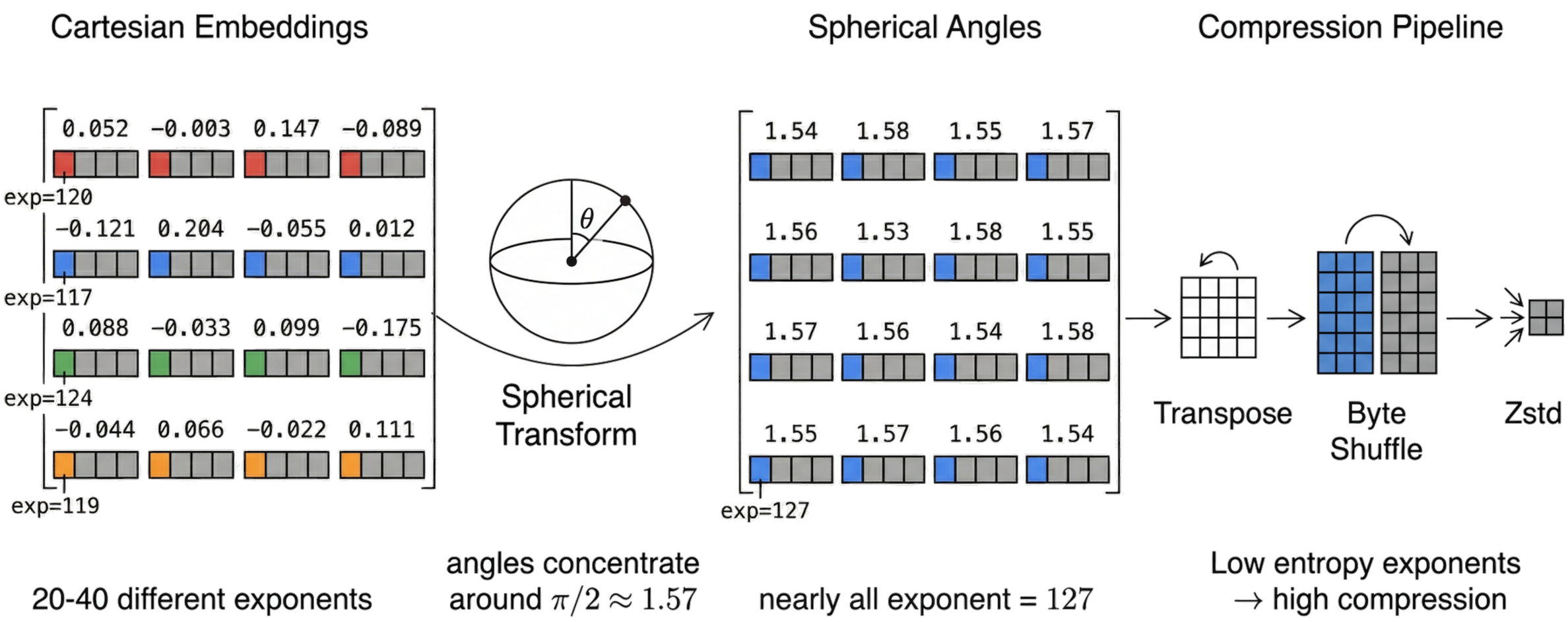
Han Xiao Jina AI by Elastic han.xiao@jina.ai

jzip C impl. arXiv:2602.00079

ICLR International Conference On Learning Representations

1.58× compression · Error < float32 ϵ · Zero retrieval degradation · Training-free

How It Works



Introduction

Embedding vectors power RAG pipelines, agentic search, and multimodal retrieval. A typical model produces 1024d float32 vectors at 4 KB each – 100M embeddings require 400 GB. ColBERT produces one embedding per token, multiplying storage $\sim$100×.
The state-of-the-art lossless approach (ZipNN, Blosc) transposes the embedding matrix, byte-shuffles to group exponents, and applies entropy coding – but achieves only **1.2**×. Most models produce unit-norm vectors on S^{d-1}, yet existing methods ignore this geometric structure.

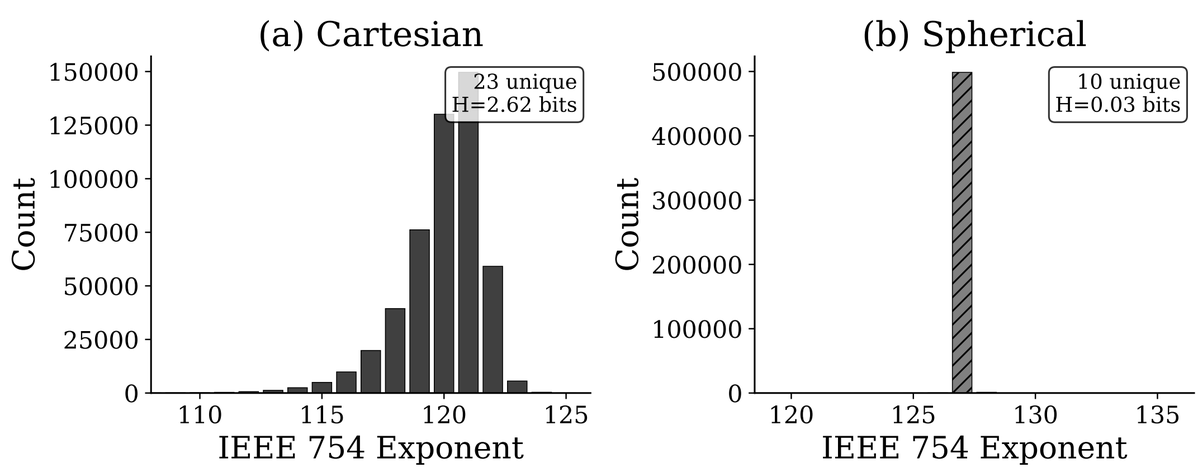
Spherical Coordinates

Unit-norm embeddings ($\|\mathbf{x}\| = 1$) lie on the hypersphere S^{d-1}. Spherical angles concentrate around $\pi/2$ in high dimensions, collapsing IEEE 754 exponents to a single value (127) with probability >0.999:

- Exponent entropy: **2.6** → **0.03** bits/byte
- Mantissa entropy: **8.0** → **4.5** bits/byte
- Combined: **1.58**× compression (+ dimension $d \to d{-}1$)

ϵ-*bounded*: reconstruction error stays below float32 ϵ (1.19×10^{-7}), 10× lower than mantissa truncation. Fills the gap between lossless 1.2× and lossy 4×+. Throughput: **487 MB/s** encode, **605 MB/s** decode.

Exponent Distribution



jina-embeddings-v4 (2048d). Cartesian: 23 exponents → Spherical: 99.7% at exponent 127.

Algorithm

Compress($\mathbf{X} \in \mathbb{R}^{n \times d}$): ▷ $\mathbf{X}$ unit-norm
  $\mathbf{\Theta} \leftarrow \texttt{ToSpherical}(\mathbf{X})$ ▷ $n \times (d{-}1)$ angles
  $\mathbf{T} \leftarrow \texttt{Transpose}(\mathbf{\Theta})$ ▷ Group same positions
  $\mathbf{B} \leftarrow \texttt{ByteShuffle}(\mathbf{T})$ ▷ Separate exp/mantissa
  return $\texttt{Zstd}(\mathbf{B})$

Decompress($\mathbf{C}$):
  $\mathbf{B} \leftarrow \texttt{ByteUnshuffle}(\texttt{Zstd}^{-1}(\mathbf{C}))$
  $\mathbf{\Theta} \leftarrow \texttt{Transpose}(\mathbf{B})$

Similarity($\boldsymbol{\theta}, \boldsymbol{\phi}$): ▷ Direct cosine
  $R \leftarrow \cos(\theta_{d-1} - \phi_{d-1})$
  for $k = d{-}2, \ldots, 1$ **do**
    $R \leftarrow \cos\theta_k \cos\phi_k + \sin\theta_k \sin\phi_k \cdot R$
  return R ▷ $= \mathbf{x} \cdot \mathbf{y}$

Direct Cosine from Spherical Angles

Cosine similarity can be computed directly from spherical angles without reconstructing Cartesian coordinates. The SIMILARITY procedure computes $\mathbf{x} \cdot \mathbf{y}$ from angles $(\theta_1, \ldots, \theta_{d-1})$ and $(\phi_1, \ldots, \phi_{d-1})$ via a backward recurrence in $O(d)$ operations, derived by expanding the Cartesian dot product in spherical form and factoring the cumulative sine products. This enables **streaming similarity during decompression**, early termination in top-k retrieval, and fused GPU kernels – without materializing the full Cartesian vector.

1.58× Compression Rate

Method	Size	Ratio	Max Err	Mean Err	Cos Max Err
Raw float32	59.38	1.00×	0	0	0
gzip -9	55.14	1.08×	0	0	0
brotli -11	54.52	1.09×	0	0	0
zstd -19	55.05	1.08×	0	0	0
npz	55.14	1.08×	0	0	0
fpzip	54.11	1.10×	0	0	0
zfp	58.99	1.01×	0	0	0
SZ3	55.03	1.08×	0	0	0
ZipNN (Baseline)	49.57	1.20×	0	0	0
Base+Trunc 5b	42.23	1.47×	9e-7	2e-8	2e-6
Base+Trunc 6b	40.30	1.55×	2e-6	5e-8	5e-6
Base+Trunc 7b	38.40	1.62×	4e-6	9e-8	1e-5
Spherical (Ours)	**37.59**	**1.58×**	**9e-8**	**2e-8**	**2e-7**

jina-v4, 7600 vectors, 2048d. Sizes in MB. Best ratio *and* lowest error.

Text, Image, Multi-Vector Embeddings

Model	Dim	Raw	Baseline	Ours	Ratio	Impr.
Text Embeddings						
MiniLM	384	11.13	9.37	7.43	1.50×	+26.0%
E5-small	384	11.13	9.10	7.31	1.52×	+24.5%
GTE-small	384	11.13	9.19	7.29	1.53×	+26.0%
BGE-base	768	22.27	18.60	14.61	1.52×	+27.3%
E5-base	768	22.27	18.19	14.31	1.56×	+27.2%
GTE-base	768	22.27	18.33	14.30	1.56×	+28.2%
MPNet	768	22.27	18.76	14.56	1.53×	+28.9%
Nomic-v1.5	768	22.27	18.57	14.58	1.53×	+27.4%
EmbedGemma	768	22.27	18.72	14.82	1.50×	+26.3%
jina-code-small	896	25.98	21.89	17.07	1.52×	+28.2%
jina-embeddings-v3	1024	29.69	24.95	19.81	1.50×	+26.0%
jina-clip-v2 (text)	1024	29.69	24.97	20.03	1.48×	+24.6%
BGE-large	1024	29.69	24.85	19.36	1.53×	+28.4%
E5-large	1024	29.69	24.32	18.94	1.57×	+28.4%
mE5-large	1024	29.69	24.32	18.91	1.57×	+28.6%
GTE-large	1024	29.69	24.34	18.85	1.58×	+29.0%
Qwen3-Embed-0.6B	1024	29.69	24.94	19.52	1.52×	+27.8%
BGE-M3 (text)	1024	29.69	24.91	19.38	1.53×	+28.6%
jina-code-large	1536	44.53	37.48	28.40	1.57×	+32.0%
jina-embeddings-v4	2048	39.06	32.44	24.61	1.59×	+31.8%
Multimodal Image						
jina-clip-v1	768	5.86	4.90	3.88	1.51×	+26.5%
jina-clip-v2	1024	7.81	6.52	5.22	1.50×	+24.9%
jina-embeddings-v4	2048	15.63	12.95	9.84	1.59×	+31.6%
Multi-Vector ColBERT						
jina-embeddings-v4	128	27.70	22.69	18.82	1.47×	+20.5%
jina-colbert-v2	1024	243.22	202.96	160.48	1.52×	+26.5%
BGE-M3	1024	239.89	197.87	154.13	1.56×	+28.4%

1.47–1.59× across all configs. Zero retrieval degradation. Sizes in MB.

Relation to TurboQuant

TurboQuant (ICLR 2026) independently identifies the same geometric phenomenon: angle concentration on the hypersphere, exploiting it for lossy quantization at 4×+. The convergence of three independent groups (TurboQuant, PolarQuant, and this work) validates angle concentration as a **fundamental geometric property**. ECF8/DFloat11 exploit *natural* concentration in weights; our method *creates* it via deterministic geometric transform.