Quantization Aware Matryoshka Adaptation: Leveraging Matryoshka Learning, Quantization, and Bitwise Operations for Reduced Storage and Improved Retrieval Speed

Anonymous Author(s)

ABSTRACT

We introduce *Quantization Aware Matryoshka Adaptation (QAMA)*, a unified framework for creating compact yet semantically rich embeddings through **Matryoshka Representation Learning** and multi-level quantization. Our approach learns nested embeddings that gracefully shrink to smaller dimensional subsets and leverages bitwise operations (XOR, NOT, POPCOUNT) for efficient retrieval. By augmenting transformer-based encoders with lightweight feedforward layers and specialized regularization (Matryoshka Loss, Orthogonality, Information Bottleneck, and Quantization Loss), we produce quantization-friendly representations that preserve essential information in early dimensions.

We explore 0.5-bit, 1-bit, 1.5-bit, and 2-bit quantization levels, as well as a Hybrid Quantization scheme that adaptively allocates higher precision to dimensions with greater information content. Across extensive evaluations on Modern BERT (MB) and MiniLM models, **2-bit quantization** and **Hybrid Quant** consistently recover up to 95–98% of the original full-precision (FP32) performance while reducing memory usage by over 90%. Even at low embedding dimensions (e.g., 96–192), QAMA's hierarchical training ensures that performance remains surprisingly robust, highlighting the effectiveness of our bit-level expansions and nested representation learning.

Our proposed end-to-end training with quantization-aware loss yields embeddings that map cleanly into discrete levels, supporting rapid Hamming distance calculations for semantic similarity search. Our framework **QAMA** offers a practical means to optimize embedding storage and retrieval speed for information retrieval systems.

CCS CONCEPTS

• **Information systems** → *Information retrieval*; *Query representation*; **Novelty in information retrieval**; *Language models.*

KEYWORDS

embedding compression, embedding quantization, information retrieval, hierarchical embeddings

ACM Reference Format:
Anonymous Author(s). 2025. Quantization Aware Matryoshka Adaptation: Leveraging Matryoshka Learning, Quantization, and Bitwise Operations for Reduced Storage and Improved Retrieval Speed. In *Woodstock '25: ACM Symposium on Neural Gaze Detection, June 03–05, 2025, Woodstock, NY*. ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Semantic embeddings serve as foundational building blocks for information retrieval systems, transforming text into high-dimensional vectors that capture semantic relationships. However, they come with substantial computational and storage costs that pose deployment challenges, particularly for resource-constrained environments or large-scale retrieval systems.

Traditional approaches to reduce embedding size, such as post-training quantization [21] or dimension pruning [29], often lead to significant degradation in retrieval quality. Knowledge distillation [15] and parameter-efficient methods [18] offer partial solutions but struggle to maintain performance under aggressive compression.

Traditional approaches to reduce embedding size, such as post-training quantization [21], dimension pruning [29], Knowledge distillation [15] and parameter-efficient methods [18] struggle to maintain performance under aggressive compression and lead to significant degradation in retrieval quality. Even advanced techniques like Product Quantization [23] or binary embeddings [43, 45, 48] face challenges in preserving fine-grained semantic relationships. The fundamental challenge lies in maintaining semantic fidelity while drastically reducing both storage requirements and similarity computation costs.

Matryoshka Representation Learning (MRL) [27] offers a promising direction by creating hierarchical embeddings where smaller representations are nested within larger ones. MRL aims to concentrate essential semantic information in early dimensions, allowing for flexible dimension selection without retraining. However, existing MRL approaches lack explicit mechanisms to enforce this nesting property and often struggle when combined with aggressive compression techniques.

In this paper, we present **Quantization Aware Matryoshka Adaptation (QAMA)**, a unified framework that addresses these challenges through three key innovations:

(1) **Hierarchical Information Organization:** We enhance existing models with lightweight feedforward layers and specialized loss functions (Matryoshka Loss, Orthogonality, Information Bottleneck) that actively concentrate essential semantic information in early dimensions. Our method shapes the embedding space during training to ensure effective dimensional reduction.

Conference acronym 'XX, June 03–05, 2025, Woodstock, NY
© 2025 Association for Computing Machinery.
ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

(2) **Trainable Multi-Level Quantization:** We introduce an end-to-end approach for ultra-low bit quantization (0.5-bit to 2-bit per dimension) that learns optimal quantization thresholds while keeping embeddings quantization-friendly. Unlike previous methods [43, 45], our approach maintains fine-grained similarity distinctions even at extremely low bit widths through carefully designed codebook expansions and loss functions.

(3) **Hybrid Precision Architecture:** Rather than applying uniform quantization, we allocate bits based on information content - using higher precision for critical early dimensions while progressively reducing precision for later dimensions. This novel approach aligns storage precision with semantic importance, offering better compression-accuracy trade-offs than traditional approaches [22, 41].

Our specialized loss functions actively shape the embedding space during training. The Matryoshka Loss ensures information hierarchy, while Orthogonality and Information Bottleneck regularizations prevent redundancy and encourage compact representations. This is further enhanced by Adaptive Variance Control which prevents degenerate embeddings. To enable efficient similarity search, we extend the use of **Hamming distance** to our multi-level quantized embeddings through careful bit-level expansions. By mapping quantized vectors to optimized binary representations, we enable fast retrieval using hardware-accelerated bitwise operations (XOR, POPCOUNT). This approach significantly improves retrieval speed compared to traditional floating-point calculations, making our method particularly suitable for large-scale applications.

Our extensive experiments demonstrate that QAMA achieves substantial storage reduction while maintaining competitive accuracy across different models and scenarios. For instance, our hybrid quantization approach reduces storage requirements by over 90% while preserving more than 95% of the original retrieval accuracy.

2 RELATED WORK

Large-scale language models and high-dimensional embeddings have greatly advanced Natural Language Processing (NLP) and Information Retrieval (IR). However, they often demand substantial storage and compute. Various compression techniques have thus emerged to reduce size while retaining performance. This section discusses embedding approaches in IR, combined strategies for compressing large language models (LLMs) and embeddings, and Matryoshka Representation Learning (MRL).

Embedding Models in Information Retrieval. Early word embeddings like Word2Vec [33] and GloVe [38] paved the way for dense vector representations. Subsequent transformers, including BERT [8], RoBERTa [30] and Sentence-BERT [39], introduced contextual embeddings with higher accuracy but also higher memory and compute costs. Recent research prioritizes compact architectures for deployment efficiency.

Compression Techniques for Language Models and Embeddings. Han et al. [13] proposed *Deep Compression*, combining pruning, quantization, and coding. The *Lottery Ticket Hypothesis* [11] highlights subnetworks that can perform on par with their original dense models. Knowledge distillation [15, 24, 42] transfers capability from large "teacher" models to smaller "student" models. Recent approaches increasingly integrate compression awareness during training [44, 55]. Our work extends this line by introducing quantization and dimensional hierarchy constraints during the learning phase.

Both post-training [21] and quantization-aware training [19, 34] constrain numerical precision. LLM.int8() [7] and BitNet [52] demonstrate successful quantization for large language models. Binary and ternary embeddings [43, 45] operate efficiently via bitwise operations. Product Quantization [23] partitions embeddings into sub-vectors for approximate nearest neighbor search. Mixed-precision approaches [10, 54] allocate different bit-widths based on layer sensitivity. In contrast, we propose dimension-wise precision allocation guided by information content

Methods like PCA [25], SVD [12], and autoencoders [16] systematically compress embeddings, while t-SNE [49] and UMAP [32] focus primarily on visualization by preserving local neighborhoods at the expense of global structure and are unsuitable for information retrieval tasks due to their non-parametric nature and inability to process new data without recomputation. In contrast, our approach creates hierarchical, nested embeddings through explicit loss functions that concentrate essential semantic information in early dimensions while maintaining cross-scale consistency, enabling dynamic dimension selection without retraining and preserving both local and global relationships crucial for retrieval tasks. Pruning low-variance dimensions [29] can yield more compact models with minimal loss in semantic fidelity, but lacks the systematic information organization and quantization awareness of our approach.

Efficient Similarity Computation. Recent advances in approximate nearest neighbor search have leveraged binary codes [53] and locality-sensitive hashing [1, 2]. Our work contributes novel bit-level expansions (e.g., 2-bit to 3-bit mapping) that preserve fine-grained similarities while enabling efficient XOR and POPCOUNT operations. This bridges the gap between ultra-low-bit quantization and accurate similarity preservation.

Matryoshka Representation Learning. Matryoshka Representation Learning (MRL) [27] aims to create hierarchical embeddings where smaller embeddings are nested within larger ones, analogous to Russian nesting dolls. This approach allows models to adaptively use embeddings of different sizes based on resource constraints, without the need for retraining. MRL focuses on generating these hierarchical embeddings such that truncated dimensions can still yield functional representations. While prior works have attempted to promote such nesting properties, they typically did so without explicit loss functions tailored to enforce them. We incorporate three explicit *Losses to Promote Matryoshka Property* to concentrate key information in early dimensions.

Our Approach in Context. While these prior approaches have made significant advances in embedding compression and efficient retrieval, they typically address individual aspects of the problem in isolation. Post-training quantization methods [21] often struggle with accuracy degradation, while MRL [27] lacks explicit mechanisms for ultra-low-bit representations. Similarly, current mixed-precision techniques [10] focus on layer-wise quantization rather than dimension-wise precision allocation that aligns with semantic importance. More recent approaches within Hugging Face, Jina AI and Vespa AI [6, 26, 58] have also focused on MRL and quantization, where [26] has proposed to integrate MRL with quantization, but their training process doesn't adapt older models and doesn't support hybrid precision allocation. Our framework QAMA unifies and extends these directions through three key innovations, namely *Quantization-aware Matryoshka Learning*, *Hybrid Precision Allocation*, and *Efficient Bit-wise Similarity Computation.*

3 METHODOLOGY

Our approach combines multiple techniques to create efficient and compact embeddings while preserving semantic relationships. The key components include:

- Neural transformation layers for improved quantization and matryoshka representation learning
- Hybrid architecture combining different quantization levels
- Loss functions to promote quantization and matryoshka representation learning
- Efficient bit-packing and similarity computation

3.1 Quantization Framework

Our quantization framework converts high-precision embeddings into compact, discrete representations while preserving semantic relationships through a combination of neural transformations and adaptive threshold-based discretization. We first apply a feedforward transformation $\mathbf{z} = \text{FFN}(\mathbf{x}; \theta)$ to optimize the embedding space for quantization, where $\mathbf{x}$ is the input embedding and θ represents trainable parameters. We implement multiple quantization levels for flexible storage-performance trade-offs: 2-bit (4 levels), 1.5-bit (3 levels), 1-bit (2 levels), and 0.5-bit (combines dimension pairs) quantization. For k-bit quantization with 2^k levels, we use $(2^k - 1)$ learnable thresholds per dimension to partition the continuous embedding space. For instance, 2-bit quantization employs three thresholds $\{\theta_1, \theta_2, \theta_3\}$ to create four bins, while 1-bit quantization uses a single threshold for binary partitioning. For a d-dimensional embedding space $\mathbf{X} \in \mathbb{R}^{N \times d}$, we initialize thresholds using percentile statistics to ensure balanced quantization: $\theta_{i,j} = \text{Quantile}\big(\{x_{1,j}, \ldots, x_{N,j}\},\ q_i\big)$ where q_i are evenly-spaced quantiles in $(0, 1)$. During training, thresholds adapt via exponential moving average:

$$\theta_{i,j}^{(\text{new})} = \mu\, \theta_{i,j}^{(\text{old})} + (1 - \mu)\ \widehat{\theta}_{i,j}^{(\text{batch})}$$

with momentum coefficient $\mu \in [0, 1]$ and batch-computed thresholds $\widehat{\theta}_{i,j}^{(\text{batch})}$. The complete process for an input embedding involves: (1) applying the trainable transformation $\mathbf{z} = \text{FFN}(\mathbf{x})$, (2) normalizing $\hat{\mathbf{z}} = \text{normalize}(\mathbf{z})$, and (3) assigning discrete codes: $q_j = \sum_{i=1}^{2^k-1} \mathbb{1}[\hat{z}_j > \theta_{i,j}]$

End-to-End Trainability. Our framework achieves end-to-end trainability by optimizing continuous-valued embeddings while simultaneously guiding them toward quantization-friendly distributions. Rather than backpropagating through non-differentiable quantization operations, we apply quantization regularization losses (Section 3.3.3) that encourage embeddings to cluster away from quantization boundaries. The thresholds themselves are updated using statistics from the continuous embeddings.

3.2 Promoting Information High Density in Early Dimensions

Matryoshka Representation Learning (MRL) creates embeddings that maintain effectiveness when truncated. Unlike traditional embeddings where key features span all dimensions, our enhanced MRL approach concentrates critical information in early dimensions through controlled nesting.

Nested Representation Generation. Given an input embedding $\mathbf{x} \in \mathbb{R}^d$, we generate a series of nested representations at different dimension levels $\mathcal{D} = \{d_1, d_2, ..., d_K\}$ where $d_1 < d_2 < ... < d_K = d$. First, we transform the input through a feed-forward network: $\mathbf{z} = \text{FFN}(\mathbf{x}; \theta) \in \mathbb{R}^d$

Unlike traditional MRL implementations that use separate projections for each scale, we employ an efficient progressive slicing mechanism. We create level-specific representations through progressive slicing:

$$\mathbf{z}_k = \text{Slice}_{d_{k-1}}^{d_k}(\mathbf{z}), \quad k = 1, \ldots, K \tag{1}$$

The key nesting property ensures that smaller representations are perfect subsets of larger ones: $\mathbf{z}_i[1 : d_j] = \mathbf{z}_j \quad \text{for } j < i$ The full embedding at level k is constructed through concatenation: $\mathbf{e}_k = [\mathbf{z}_1; \mathbf{z}_2; ...; \mathbf{z}_k]$ We implement the transformation function FFN using the same lightweight yet effective feed-forward network as introduced in the quantization framework, where $\mathbf{z} = \text{FFN}(\mathbf{x}; \theta) = \mathbf{W}_2(\text{GELU}(\mathbf{W}_1\mathbf{x}))$. This approach inherently maintains the nested property without requiring explicit constraints, reducing computational overhead.

Combined Loss Function. Our training objective combines multiple loss components, each with their corresponding weights, applied at each dimension level:

$$\mathcal{L}_{\text{total}} = \sum_{d \in \mathcal{D}} \sum_{l \in \mathcal{L}} w_l^{(d)} \mathcal{L}_l^{(d)} \tag{2}$$

where $w_l^{(d)}$ represents the weight for loss component l at dimension level d.

3.3 Loss Functions and Training Objectives

In our proposed Matryoshka Representation Learning (MRL) framework, we integrate both *similarity preservation* and *matryoshka adaptation* losses to achieve nested, multilevel embeddings that maintain semantic quality at each quantization level. Additionally, a *quantization regularization* component ensures that the learned representation is readily quantizable with minimal loss in accuracy.

3.3.1 Similarity Preservation.

Direct Similarity MSE Loss. We first ensure preservation of pairwise similarities via a mean squared error (MSE) loss on the normalized similarity matrices:

$$\mathcal{L}_{\text{sim}}(\mathrm{X}, \mathrm{Y}) = \|\mathrm{S}_X - \mathrm{S}_Y\|_F^2 \tag{3}$$

where $\mathrm{S}_X = \text{normalize}(\mathrm{X})\ \text{normalize}(\mathrm{X})^T$ and $\mathrm{S}_Y = \text{normalize}(\mathrm{Y})\ \text{normalize}(\mathrm{Y})^T$ preserving the cosine similarities between all pairs of embeddings $\mathbf{X}$ and $\mathbf{Y}$ at different quantization levels.

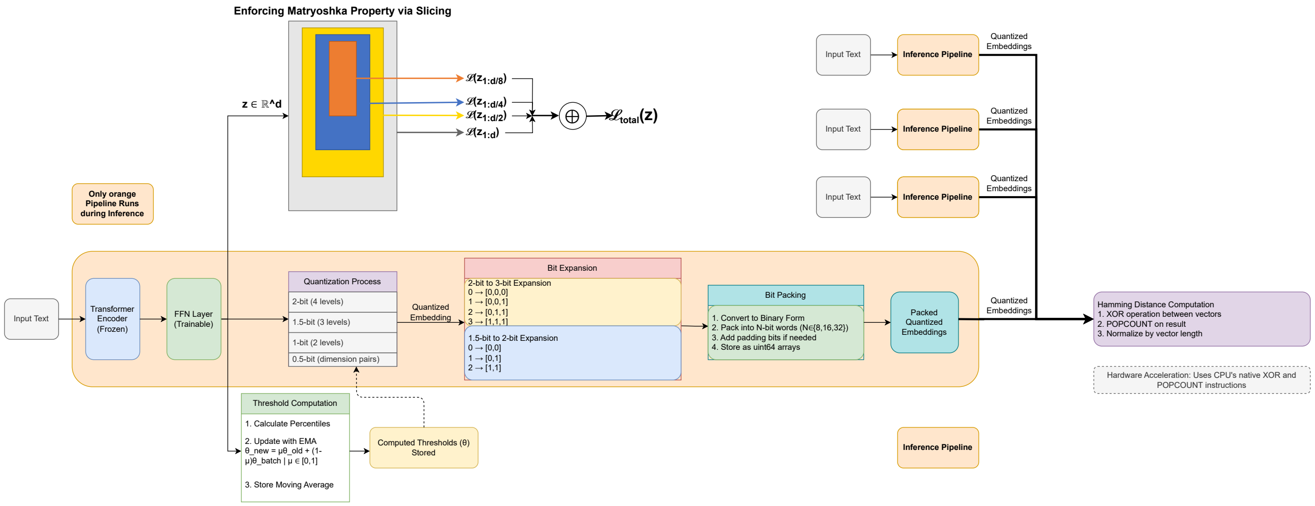

Figure 1: Overall architecture of QAMA system showing the transformation, quantization, and bit-packing stages.

KL Divergence Loss. To align the probability distributions of similarities, we employ a symmetric KL divergence with temperature scaling:

$$\mathcal{L}_{\text{kl}} = \tfrac{1}{2}\left[D_{\text{KL}}(\mathbf{P}_X \| \mathbf{P}_Y) + D_{\text{KL}}(\mathbf{P}_Y \| \mathbf{P}_X)\right], \tag{4}$$

where τ controls the sharpness of the distribution and $\mathbf{P}_X = \text{softmax}(\mathbf{S}_X/\tau)$ and $\mathbf{P}_Y = \text{softmax}(\mathbf{S}_Y/\tau)$

Rank Preservation Loss. To maintain the *relative ordering* of similarities, we implement a differentiable ranking loss:

$$\mathcal{L}_{\text{rank}} = \sum_{i,j,k} \sigma\big((s_{ij}^X - s_{ik}^X)\cdot t\big) \odot \sigma\big((s_{ij}^Y - s_{ik}^Y)\cdot t\big), \tag{5}$$

where $\sigma(\cdot)$ is the sigmoid function, t sharpens the sigmoid, and s_{ij}^X, s_{ij}^Y denote pairwise similarities in the original and quantized embeddings respectively.

Contrastive Learning. We also incorporate contrastive losses with positive and negative pairs:

$$\mathcal{L}_{\text{contrast}} = -\log \frac{\exp(s_p/\tau)}{\exp(s_p/\tau) + \sum_n \exp(s_n/\tau)}, \tag{6}$$

where s_p and s_n represent the similarities for positive and negative pairs, respectively.

3.3.2 Losses to Promote Matryoshka Property.

To further support the *Matryoshka* property (i.e., nested and progressively expanding embeddings), we introduce three *matryoshka adaptation* losses that manipulate the distribution and uniqueness of information at each dimension level.

Progressive Information Bottleneck. We encourage essential features to concentrate in earlier dimensions:

$$\mathcal{L}_{\text{ib}} = \sum_{i=1}^{n} w_i \left(\frac{|\mathbf{x}_i|^2}{x_0 + |\mathbf{x}_i|}\right)^{\alpha}, \tag{7}$$

where $w_i = i/n$ progressively increases with dimension index i, $x_0 \approx 0.1$ is a small suppression threshold, and $\alpha \approx 0.3$ controls the gradient strength near zero. This approach pushes less important information (values close to zero) towards zero in higher dimensions.

Inter-level Orthogonal Information Encoding. We promote uniqueness of newly added dimensions by enforcing orthogonality with respect to previously formed dimensions:

$$\mathcal{L}_{\text{orth}} = \sum_{l=2}^{L} \sum_{j=1}^{l-1} \|\Delta_l^T \cdot \text{normalize}(\mathbf{E}_j)\|_F, \tag{8}$$

where $\mathbf{\Delta}_l$ indicates the newly added dimensions at level l, and $\mathbf{E}_j$ consists of all dimensions up to level j. This encourages each new level of dimensions to encode novel information rather than rediscovering existing patterns.

Adaptive Variance Control. To prevent embedding collapse while allowing selective dimension suppression, we employ a slowly increasing variance penalty:

$$\mathcal{L}_{\text{var}}(t) = \max\left(0.2, \frac{e^{t/T} - 1}{e - 1}\right) \sum_{d=1}^{D} \exp(-\sigma_d), \tag{9}$$

where t is the current training step, T is the total training steps, and σ_d is the standard deviation of dimension d. This loss starts with a weak penalty and grows over time, preventing early collapse and encouraging dimensions that truly matter to preserve sufficient variance.

3.3.3 Quantization Regularization Loss.

While the above mechanisms reinforce nested representation properties, we also require a loss term to facilitate *quantization readiness*. This term discourages embeddings from clustering near quantization thresholds and encourages them to lie closer to valid quantized levels.

Formulation. Let θ be a quantization threshold for 1-bit quantization, or $\{\theta_1, \theta_2, \theta_3\}$ be thresholds for 2-bit quantization. Denoting each embedding value as x, our quantization regularization can be written as:

$$\mathcal{L}_{\text{quant}} = w(t)\Big[\exp\big(-(\min_i |x-\theta_i|)\big) + \lambda\, L_{\text{range}}(x)\Big], \quad (10)$$

where $w(t)$ progresses from 0.2 to 1.0 over training to gradually strengthen quantization constraints, and $L_{\text{range}}(x)$ enforces that x lies within a valid range (e.g., $[0,1]$ or another permissible interval). Specifically,

$$L_{\text{range}}(x) = \text{ReLU}(l-x)^2 + \text{ReLU}(x-u)^2, \quad (11)$$

where l and u are the lower and upper bounds of acceptable ranges. Intuitively, this loss repels the embedding away from threshold boundaries (to avoid ambiguity) and adjusts overly large or negative values back into a valid quantization domain.

By combining these losses, we achieve a robust training procedure where $\mathcal{L}_{\text{sim}}$, $\mathcal{L}_{\text{kl}}$, $\mathcal{L}_{\text{rank}}$, $\mathcal{L}_{\text{contrast}}$ ensure semantic fidelity across different quantization levels, $\mathcal{L}_{\text{ib}}$, $\mathcal{L}_{\text{orth}}$, $\mathcal{L}_{\text{var}}$ organize the learned features hierarchically, pushing essential features into early dimensions while maintaining uniqueness and stable variance distribution across expanding dimensional levels, and $\mathcal{L}_{\text{quant}}$ directly encourages embeddings to settle into regions that are easily separable by the quantizer, thereby reducing errors from threshold-based discretization. Finally, these losses can be summed with appropriate weights to form the complete training objective.

3.4 Hybrid Architecture

We employ a hierarchical scheme that applies different quantization levels to different fractions of the embedding dimensions based on their information content:

- **First 25% (highest information):** 2-bit quantization (expanded to 3-bit codes)
- **Next 25% (medium-high):** 1.5-bit quantization (3 levels, expanded to 2-bit codes)
- **Next 25% (medium-low):** 1-bit quantization (direct binary)
- **Final 25% (lowest):** 0.5-bit quantization (pairs of dimensions reduced via a lightweight FFN)

This arrangement yields an average of 1.625 bits per dimension, balancing storage efficiency and similarity preservation by granting higher precision to dimensions with greater semantic importance.

Dimension Reduction for 0.5-bit Encoding. For the last quarter of dimensions, pairs are combined:

$$\mathbf{z}_{\text{reduced}} = \text{FFN}_{\text{reduce}}\Big([\mathbf{x}_{2i}, \mathbf{x}_{2i+1}]\Big), \quad (12)$$

where a specialized FFN preserves essential information while halving the dimensional load.

3.5 Efficient Storage and Similarity Computation

We combine bit-packing with carefully designed expansions that preserve semantic similarity in quantized embeddings while minimizing storage overhead.

Bit Packing Process:

1. Convert quantized vectors to binary form.
2. Add padding bits if needed.
3. Store as `uint64` arrays for optimal alignment.

Binary Codebook Expansion. We expand the binary codes to preserve fine-grained similarities while minimizing storage overhead.

2-bit to 3-bit:

$$\text{Codebook}_{2\to3} = \begin{cases} 0 \longrightarrow [0,0,0] \\ 1 \longrightarrow [0,0,1] \\ 2 \longrightarrow [0,1,1] \\ 3 \longrightarrow [1,1,1] \end{cases} \quad (13)$$

Expanding 2-bit values (0–3) to 3 bits preserves relative distances more accurately, preventing similar embeddings from collapsing to identical Hamming codes. For example:

- Original 2-bit values [0,1] and [1,0] have equal Hamming distance to [1,1]
- Expanded codes [0,0,1] and [0,1,1] have different Hamming distances to [1,1,1]
- This better reflects the original quantization levels' relationships

1.5-bit to 2-bit:

$$\text{Codebook}_{1.5\to2} = \begin{cases} 0 \longrightarrow [0,0] \\ 1 \longrightarrow [0,1] \\ 2 \longrightarrow [1,1] \end{cases} \quad (14)$$

Maps three-level codes to 2 bits, improving similarity preservation while keeping compact storage.

Basic Hamming Similarity: For binary vectors $\mathbf{x}, \mathbf{y}$ of length n (bits),

$$\text{sim}_H(\mathbf{x}, \mathbf{y}) = 1 - \frac{\text{POPCOUNT}(\mathbf{x} \oplus \mathbf{y})}{n}, \quad (15)$$

where $\oplus$ is XOR and POPCOUNT counts set bits.

Implementation Details: Modern CPUs offer hardware-accelerated `XOR` and `POPCNT` instructions (optionally AVX-512) enabling high-throughput Hamming distance computation on packed 64-bit words. We leverage `numpy.packbits`, `numpy.unpackbits`, `numpy.bitwise_xor`, and `numpy.bitwise_count` for fast, vectorized bit manipulation and codebook expansions.

4 EXPERIMENTAL SETUP

We evaluated our approach using a suite of benchmark retrieval datasets from the MTEB (Massive Text Embedding Benchmark) library [35], including ArguAna [46, 50], CQADupstackTexRetrieval [17], ClimateFEVER [9], DBPedia [14], FEVER [47], FiQA2018 [46], HotpotQA [59], MSMARCO [36], NFCorpus [4], NQ (Natural Questions) [28], QuoraRetrieval [20, 46], SCIDOCS [5], SciFact [51], TREC-COVID [40, 46], and Touche2020 [3].

For our experiments, we employed two distinct Transformer models: MiniLM [39, 56], a compact 12-layer model with 384 dimensional embeddings derived through knowledge distillation, and Modern BERT (MB) [37, 57], a more recent 22-layer architecture with 768 dimensional embeddings incorporating state-of-the-art design principles. This selection allows us to evaluate our approach across different model scales and architectural generations, from efficient distilled models to modern high-capacity architectures. Demonstrating robust results on both a smaller, older architecture and a more recent, higher-capacity model indicates that practitioners can attain substantial compression benefits—via quantization and nested dimensionality reduction—across diverse Transformer-based architectures. We compared our approach against the following baselines: **Original Model (FP32)** (full-precision 32-bit floating point embeddings), **FP16** (half-precision 16-bit), **Int8** (8-bit integer quantization), and **Simple Threshold Quantization** (basic 2-bit, 1.5-bit, and 1-bit quantization using fixed thresholds without learned transformations or Matryoshka representation).

The training process optimizes our Matryoshka model $\mathcal{M}$ initialized from a pretrained model $\mathcal{E}$. For each batch, normalized base embeddings are transformed to produce both non-quantized and quantized representations at different dimension levels. The training objective combines multiple weighted loss terms as detailed in Section 3.3. Quantization thresholds are initialized using percentile-based statistics and updated with momentum. We used AdamW [31] optimizer with $\eta = 1 \times 10^{-4}$, warm-up scheduling, and gradient clipping, training for 5 epochs. For evaluation, we used NDCG@10 which measures ranking quality by comparing the relevance scores of retrieved documents against an ideal ranking, normalized to [0,1].

5 RESULTS AND ANALYSIS

We evaluate the performance across various embedding dimensions and quantization levels to demonstrate the effectiveness of our methods in reducing storage requirements and improving retrieval speed while maintaining high accuracy.

Tables 1 and 2 compare the performance of Modern BERT (MB) and MiniLM at various quantization levels and dimensions, using nDCG@10 average across multiple datasets as the primary metric.

Key Insights:

- **2-bit Quantization Nears FP32 Performance:** As shown in the ablation studies, 2-bit quantization retains around 95–98% of FP32 performance, demonstrating its effectiveness in preserving semantic information.
- **Hybrid Quantization Efficiency:** The Hybrid approach offers additional storage savings while maintaining performance close to 2-bit, as evidenced by the ablation results.
- **Role of Matryoshka Loss:** The introduction of Matryoshka Loss significantly enhances performance, particularly at lower dimensions, by concentrating essential information in early dimensions.

Table 1: Performance of Modern BERT (MB) with different quantization levels and embedding dimensions.

Dimension	1-bit	1.5-bit	Hybrid Quant	2-bit	FP32
768	0.4381	0.4536	0.4680	0.4687	0.4720
384	0.4167	0.4429	0.4509	0.4593	0.4695
256	0.3691	0.4228	0.4465	0.4513	0.4680
192	0.3285	0.3901	0.4245	0.4327	0.4512
96	0.2908	0.3455	0.3850	0.3919	0.4247

Table 2: Performance of MiniLM with different quantization levels and embedding dimensions.

Dimension	1-bit	1.5-bit	Hybrid Quant	2-bit	FP32
384	0.3839	0.4101	0.4160	0.4185	0.4286
192	0.3724	0.3923	0.4017	0.4109	0.4219
128	0.3571	0.3814	0.3865	0.3917	0.3963
96	0.3417	0.3649	0.3695	0.3712	0.3792
48	0.2687	0.2871	0.2919	0.2897	0.3014

Note that for Modern BERT, with dimension FP32, and dimension as 768 (original dimension), the NDCG@10 is 0.4720 and this was using the original model with no changes. For MiniLM, with dimension FP32, and dimension as 384 (original dimension), the NDCG@10 is 0.4286 and this was using the original model with no changes. From the results, we observe that our proposed QAMA framework achieves competitive performance compared to the full-precision (FP32) embeddings, even at significantly reduced storage sizes. For instance, the 2-bit quantized embeddings at 384 dimensions achieve an nDCG@10 of 0.4593 with Modern Bert and 0.4185 with MiniLM, which is close to the FP32 performance while reducing the storage requirements substantially. Although slightly behind pure 2-bit in accuracy, Hybrid Quant offers additional storage savings (≈ 1−4% more than 2-bit) while keeping performance within 1−2% of the best quantized results.

Our results show that increasing the quantization level (from 1-bit to 2-bit) consistently improve performance, highlighting the trade-off between storage efficiency and model accuracy. The Hybrid Quantization approach, which uses different quantization levels for different embedding dimensions, achieves performance close to 2-bit quantization while offering better storage efficiency (typically within about 1–2% of the 2-bit nDCG@10). For example, at 384 dimensions in Modern BERT, Hybrid Quant yields 0.4509 nDCG@10 compared to 0.4593 for 2-bit, recovering about 98.2% of the latter's performance, while saving approximately 2% additional storage.

5.1 Effect of Embedding Dimensions

Figure 2 show that higher dimensions generally yield stronger performance due to increased representational capacity. However, our methods maintain competitive accuracy at lower dimensions. In particular, the ablation results highlight that, at around 192 dims for MB or 96 dims for MiniLM, the Matryoshka Loss and Orthogonality Regularization become increasingly critical (see Table 4 and Table 6). For example, MB at 192 dims with 2-bit quantization retains 0.4327 nDCG@10, compared to 0.4512 for FP32 (i.e., around 96% of the full-precision performance). At lower dimensions, Matryoshka Loss becomes critical in preserving performance by ensuring hierarchical information encoding. These results illustrate that while higher dimensions generally improve performance, our proposed methods enable lower-dimensional embeddings to achieve competitive results, which is beneficial for resource-constrained environments.

5.2 Ablation Studies

To evaluate the contribution of each component in our proposed methodology, we perform ablation studies on the MB and MiniLM models at two embedding dimensions each. Specifically, we consider

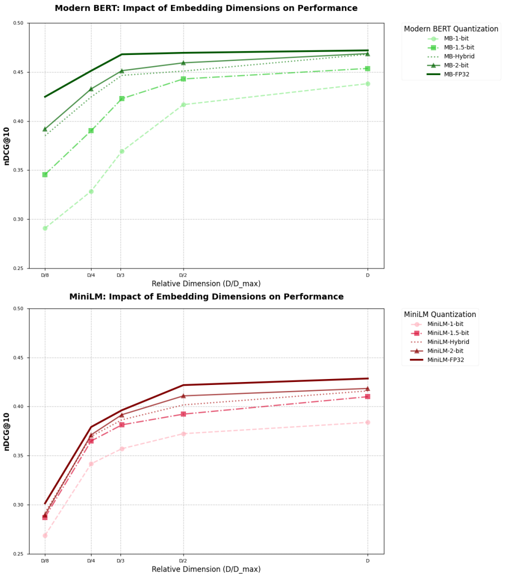

Figure 2: Impact of embedding dimensions on performance for Modern BERT (MB) and MiniLM model across different quantization levels.

dimensions 384 and 192 for the MB model, and dimensions 192 and 96 for the MiniLM model. Tables 3 and 4 present the ablation results for the MB model, while Tables 5 and 6 show the results for the MiniLM model. We highlight a few *important* observations here:

- **Matryoshka Loss Yields Large Jumps in Lower Dimensions:** E.g., for MiniLM at 96 dims, final performance doubles from *Thresholds Only* to *+ MVC + Orthogonality + IB* (Table 6). This underscores that hierarchical training is essential when dimension is aggressively reduced.
- **Adaptive Variance Control as a Final "Booster" Step:** In each table, AVC tends to add a 2–4% relative improvement. This mechanism is vital for preventing degenerate embeddings in scenarios with both low bits and low dimensions.
- **MB vs. MiniLM:** MB generally retains better performance at higher dimensions, while MiniLM benefits significantly from the matryoshka methods, indicating the approach's adaptability across different architectures.
- **Sensitivity to Compression:** MB shows resilience to dimensional downscaling, whereas MiniLM exhibits faster trade-off declines, reflecting differences in model architecture.

Table 3: Ablation study for MB model at 384 dimensions.

Training Components	1-bit	1.5-bit	Hybrid Quant	2-bit	FP32
Thresholds Only	0.3900	0.4250	0.4358	0.4370	-
+ Trainable FFN Transform	0.3922	0.4275	0.4380	0.4390	0.4410
+ Quantization Loss	0.3945	0.4298	0.4402	0.4415	-
+ Matryoshka Loss	0.3998	0.4318	0.4410	0.4425	0.4632
+ Orthogonality Regularization	0.4012	0.4329	0.4430	0.4440	0.4645
+ Information Bottleneck Regularization	0.4025	0.4340	0.4442	0.4455	0.4658
+ Adaptive Variance Control	**0.4167**	**0.4429**	**0.4509**	**0.4593**	**0.4695**

Table 4: Ablation study for MB model at 192 dimensions.

Training Components	1-bit	1.5-bit	Hybrid Quant	2-bit	FP32
Thresholds Only	0.2900	0.3550	0.3755	0.3850	-
+ Trainable FFN Transform	0.2922	0.3575	0.3778	0.3872	0.3900
+ Quantization Loss	0.2943	0.3597	0.3801	0.3893	-
+ Matryoshka Loss	0.3107	0.3732	0.3945	0.4050	0.4405
+ Orthogonality Regularization	0.3125	0.3752	0.3964	0.4068	0.4422
+ Information Bottleneck Regularization	0.3142	0.3770	0.3983	0.4087	0.4440
+ Adaptive Variance Control	**0.3285**	**0.3901**	**0.4245**	**0.4327**	**0.4512**

Table 5: Ablation study for MiniLM model at 192 dimensions.

Training Components	1-bit	1.5-bit	Hybrid Quant	2-bit	FP32
Thresholds Only	0.2947	0.3205	0.3260	0.3324	-
+ Trainable FFN Transform	0.3112	0.3538	0.3645	0.3753	0.3841
+ Quantization Loss	0.3267	0.3661	0.3773	0.3883	-
+ Matryoshka Loss	0.3318	0.3709	0.3821	0.3922	0.3996
+ Orthogonality Regularization	0.3354	0.3741	0.3850	0.3940	0.4019
+ Information Bottleneck Regularization	0.3389	0.3768	0.3876	0.3961	0.4035
+ Adaptive Variance Control	**0.3724**	**0.3923**	**0.4017**	**0.4109**	**0.4219**

Table 6: Ablation study for MiniLM model at 96 dimensions.

Training Components	1-bit	1.5-bit	Hybrid Quant	2-bit	FP32
Thresholds Only	0.2050	0.2330	0.2375	0.2428	-
+ Trainable FFN Transform	0.2250	0.2570	0.2715	0.2860	0.3340
+ Quantization Loss	0.2640	0.3017	0.3161	0.3304	-
+ Matryoshka Loss	0.3100	0.3350	0.3450	0.3550	0.3600
+ Orthogonality Regularization	0.3150	0.3380	0.3480	0.3575	0.3625
+ Information Bottleneck Regularization	0.3175	0.3400	0.3500	0.3590	0.3640
+ Adaptive Variance Control	**0.3417**	**0.3649**	**0.3695**	**0.3712**	**0.3792**

Performance Across Dimensions and Impact of Dimensional Reduction: Our methods demonstrate remarkable resilience to dimensional reduction while maintaining competitive performance across different scales. At higher dimensions, the MB model with 2-bit quantization achieves an nDCG@10 of 0.4687 at 768 dimensions, close to the FP32 performance of 0.4720 (99.3% retained). When reducing to 384 dimensions, it maintains 0.4593 (97.3% of FP32), and even at 192 dimensions still achieves 0.4327 (96% of its corresponding FP32 score of 0.4512).

The effectiveness of our dimensional reduction approach is particularly evident in the step-wise degradation patterns. For MB at 2-bit quantization, halving dimensions from 384 to 192 results in only a 6% relative drop (0.4593 to 0.4327), while the corresponding FP32 models show a similar 3.9% decrease (0.4695 to 0.4512). This near-parallel degradation suggests our quantization effectively preserves the inherent information hierarchy across dimensions.

MiniLM exhibits similar robustness: at 2-bit quantization, performance decreases from 0.4185 at 384 dimensions to 0.4109 at 192 dimensions (just 1.8% drop), and to 0.3712 at 96 dimensions (about 10% reduction from 192 dimensions). Notably, when compared to their respective FP32 baselines, the 2-bit models consistently retain 95-97% of full-precision performance across all dimension levels (e.g., 0.3712 vs 0.3792 at 96 dimensions, 97.9% retained). In Table 6, even at aggressive 96-dimension compression, the full pipeline (including Matryoshka Loss, Orthogonality, Information Bottleneck, and AVC) achieves 0.3712 at 2-bit quantization—a remarkable improvement over the baseline 0.2428 (*Thresholds Only*) and approaching the FP32 performance of 0.3792. This represents not just preservation of performance but actual enhancement through our specialized training regime, with the 2-bit model retaining 97.9%

of FP32 performance even at this highly compressed dimension. These consistent patterns across both models and various dimension levels demonstrate that our approach successfully concentrates the most salient features in earlier dimensions while maintaining effective quantization.

Impact of Matryoshka Loss and Associated Regularizations: As shown in the ablation studies (Tables 3, 4, 5, and 6), introducing the Matryoshka Loss yields significant performance gains across all quantization levels and dimensions. For instance, in Table 4 at 192 dimensions (2-bit), the Modern BERT (MB) model's nDCG@10 score improves from 0.3850 (*Thresholds Only*) to 0.4050 upon adding Matryoshka Loss. An even larger jump can be seen for MiniLM at 96 dimensions (Table 6), where 2-bit quantization increases from 0.2428 (*Thresholds Only*) to 0.3550 after Matryoshka Loss—over a 46% relative gain.

Further improvements occur once Orthogonality and Information Bottleneck Regularizations are applied, especially at lower dimensions. For example, in Table 4 with MB at 192 dimensions (2-bit), these two regularizations boost the nDCG@10 score from 0.4050 to 0.4087 before Adaptive Variance Control is introduced, illustrating how newly added dimensions learn orthogonal (less redundant) features while penalizing unhelpful information in higher dimensions.

Adaptive Variance Control (AVC): AVC then provides an additional and sometimes substantial performance jump by preventing degenerate embedding distributions and refining the variance structure across dimensions. In Table 4 (MB at 192 dimensions, 2-bit), AVC lifts the nDCG@10 from 0.4087 to 0.4327, closing much of the gap to full-precision performance. Likewise, for MiniLM at 96 dimensions (Table 6), cumulative additions of Matryoshka Loss and other regularizations yield 0.3590 at 2-bit—a marked improvement over 0.2428 (*Thresholds Only*)—and further rise to 0.3712 with AVC.

Quantization Loss: Alongside these components, we employ a dedicated Quantization Loss that penalizes embedding values lying near threshold boundaries, thus reducing ambiguity in the quantization process. For example, in Table 6 at 2-bit, moving from *+ Trainable FFN Transform* (nDCG@10 = 0.2860) to *+ Quantization Loss* (0.3304) results in a 4.4-point absolute improvement, over 15% relative gain. By discouraging embeddings from clustering around threshold edges, the Quantization Loss ensures that discrete bins remain well-separated, minimizing quantization errors.

5.3 Storage Efficiency and Retrieval Speed

Table 7 compares the storage requirements of different quantization levels using 768-dimensional vectors using our proposed methods. Our methods offer substantial storage savings compared to full-precision embeddings. Moreover, the bitwise operations (XOR + POPCOUNT) provide different performance trade-offs for retrieval, particularly for large-scale corpora. To understand computational efficiency, we compare the relative FLOPS needed for similarity computation (with FP32 set as 1×). Additionally, to gauge real-world performance, we measure relative wall clock timing (also normalized to FP32 = 1×) by testing on various EC2 machines (across c5, m5, r5 CPU families) and then averaging the execution times over 100 runs for similarity computation between 1000 query vectors and 1M document vectors.

Table 7: Updated storage comparison for different embedding formats, with 768-dimensional vectors (top) and reduced 384-dimensional counterparts (bottom). "Relative Perf. (%)" is computed against 768-dim FP32 nDCG = 0.4720.

Format	Bits/dim	Vector Size	1M Vectors	Savings (%)	Relative Perf. (%)	Wall Clock
(Dimension = 768)						
FP32	32	3072 B	3.07 GB	0.0	100.0	1.00×
FP16 (approx)	16	1536 B	1.54 GB	50.0	99.9	0.98×
Int8 (approx)	8	768 B	768 MB	75.0	99.7	0.95×
2-bit (3-bit expanded)	3	288 B	288 MB	90.6	99.3	0.90×
1.5-bit (2-bit expanded)	2	192 B	192 MB	93.8	96.2	0.85×
1-bit	1	96 B	96 MB	96.9	92.8	0.82×
Hybrid (25% each level)	1.625	156 B	156 MB	94.9	99.2	0.87×
Reducing Embedding dimension to D/2 = 384						
FP32	32	1536 B	1.54 GB	50.0	99.5	0.70×
FP16 (approx)	16	768 B	0.77 GB	75.0	99.2	0.68×
Int8 (approx)	8	384 B	384 MB	87.5	98.1	0.65×
2-bit (3-bit expanded)	3	144 B	144 MB	95.3	97.3	0.60×
1.5-bit (2-bit expanded)	2	96 B	96 MB	96.9	93.8	0.55×
1-bit	1	48 B	48 MB	98.4	88.2	0.52×
Hybrid (25% each level)	1.625	78 B	78 MB	97.5	95.5	0.57×

The use of bitwise operations for similarity computation significantly boosts retrieval speed due to optimized CPU instructions. Our methods achieve faster retrieval times compared to traditional floating-point computations, making them suitable for real-time applications. As shown in Table 7, our quantization approaches achieve substantial memory savings with minimal degradation in performance. For 768-dimensional vectors: **2-bit quantization** reduces storage by about 90.6% (from 3.07 GB to 288 MB), while retaining 99.3% of the full-precision nDCG performance, and reduces wall clock time by a factor of 0.90× over FP32; **1-bit quantization**, although more extreme, still preserves roughly 92.8% of the performance and cuts storage by 96.9%; **Hybrid Quantization** strikes a balance, achieving 94.9% storage reduction and retaining 99.2% of full-precision performance, with this approach also modestly improving clock time to 0.87× of FP32, benefitting from bitwise operations (e.g., XOR/POPCOUNT). Reducing the embedding dimension from 768 to 384 provides substantial benefits, with **FP32 at 384 dims** saving 50% storage while retaining 99.5% nDCG and 0.70× wall clock time, and **2-bit** quantization at 384 dims achieving 95.3% storage reduction (144 MB for 1M vectors) with 97.3% performance and 0.60× speed. The more aggressive **1.5-bit** and **1-bit** options at 384 dims compress storage up to 96.9% and 98.4% respectively, though with larger performance drops to 93.8% and 88.2% of FP32, making them suitable for very memory-constrained scenarios.

5.4 Discussion

Our experiments demonstrate that QAMA achieves significant compression (90-97% storage reduction) while maintaining 95-98% of full-precision accuracy, representing a new operating point in the storage-accuracy trade-off curve. This is particularly notable given that previous approaches typically saw 10-15% accuracy drops at similar compression rates.

Our experiments reveal several key insights:

- **Matryoshka Loss Yields Large Jumps in Lower Dimensions:** E.g., for MiniLM at 96 dims, adding Matryoshka Loss improves nDCG@10 from 0.2428 to 0.3550 at 2-bit quantization (a 46% relative gain). For MB at 192 dims, it improves from 0.3850 to 0.4050 (5.2% gain).
- **Adaptive Variance Control as a Final "Booster" Step:** In each ablation table, AVC adds 2-4% relative improvement.

For MB at 384 dims with 2-bit quantization, AVC boosts nDCG@10 from 0.4455 to 0.4593 (3.1% gain).
- **Orthogonality and Bottleneck Losses:** These regularizations spread out distinct features among the newly added dimensions (Orthogonality) while forcing early dimensions to capture the most crucial semantic content (Info Bottleneck). Together they add 1-3% improvement (e.g., MB at 192 dims improves from 0.4050 to 0.4087 at 2-bit), ensuring newly added dimensions encode distinct features. The combination of Matryoshka Loss with Orthogonality Regularization appears critical - removing either causes >5% accuracy drop at high compression rates
- **Quantization Loss:** This component penalizes embeddings near threshold boundaries, reducing quantization ambiguity. For MiniLM at 96 dims, moving from *+ Trainable FFN Transform* (nDCG@10 = 0.2860) to *+ Quantization Loss* (0.3304) results in a 4.4-point absolute improvement, over 15% relative gain.
-
- **Substantial Storage and Compute Benefits:** Our approach demonstrates remarkable efficiency gains across both storage and computation. At 384 dimensions with hybrid quantization, we achieve 97.5% storage reduction (from 3.07GB to 78MB per million vectors) while maintaining 95.5% accuracy. The wall clock timing improves to 0.57× of FP32 baseline, representing a 43% speedup in computation time. These improvements scale linearly with corpus size, making our method particularly valuable for large-scale deployments.

Practical Implications: The results suggest several deployment strategies:

- For resource-constrained environments (e.g., edge devices), 1.5-bit quantization with 192 dimensions offers an excellent compromise, reducing storage by 96.9% while maintaining 93.8% accuracy.
- For latency-sensitive applications, the hybrid approach at 384 dimensions provides the best balance: 97.5% storage savings with 95.5% accuracy and only 0.57× the baseline computation time.
- When accuracy is paramount, 2-bit quantization at full dimension (768) achieves 99.3% accuracy with 90.6% storage savings.

6 CONCLUSION

Our paper introduces Quantization Aware Matryoshka Adaptation (QAMA), a unified framework that successfully addresses the dual challenges of embedding storage efficiency and retrieval speed while maintaining semantic fidelity. By combining Matryoshka Representation Learning with multi-level quantization and efficient bitwise operations, our approach achieves remarkable compression rates (90-97% storage reduction) while preserving 95-98% of full-precision accuracy—a significant improvement over previous methods that typically saw 10-15% accuracy drops at similar compression rates.

Our extensive experiments with Modern BERT and MiniLM demonstrate several key findings:

- Our proposed Matryoshka Loss and specialized regularizations (Orthogonality, Information Bottleneck) enable robust performance even under aggressive compression, with 2-bit quantization consistently recovering 95-98% of FP32 performance.
- Our hybrid quantization approach, which allocates different bit precisions based on information content, achieves 94.9% storage reduction while maintaining 99.2% accuracy and reducing computation time to 0.87× of the baseline.
- The framework shows remarkable dimensional scalability — Modern BERT maintains 96% of FP32 performance even when compressed to 192 dimensions with 2-bit quantization, while MiniLM retains 97.9% at 96 dimensions.

We show that effective embedding compression requires more than just post-processing techniques — it demands an end-to-end approach that shapes the embedding space during training. The combination of hierarchical information organization, trainable quantization thresholds, and specialized loss functions creates representations that are inherently amenable to aggressive compression while preserving semantic relationships.

Looking ahead, several promising directions emerge for future research:

- Extending QAMA to multi-modal embeddings, where different modalities might benefit from different quantization strategies
- Investigating dynamic bit allocation schemes that adapt to query patterns or computational resources in real-time
- Efficient implementation of hamming distance for multi-level quantized embeddings

Our work demonstrates that through careful design of the training pipeline and loss functions, it is possible to create highly compressed embeddings that maintain semantic fidelity while enabling faster retrieval. QAMA offers a practical solution for deploying large-scale embedding systems under real-world constraints, potentially enabling new applications in resource-constrained environments or latency-sensitive scenarios.

REFERENCES

[1] Alexandr Andoni and Piotr Indyk. 2006. Near-Optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*. 459–468. https://doi.org/10.1109/FOCS.2006.49

[2] Alexandr Andoni, Piotr Indyk, Huy L. Nguyen, and Ilya Razenshteyn. 2014. Beyond locality-sensitive hashing. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms* (Portland, Oregon) *(SODA '14)*. Society for Industrial and Applied Mathematics, USA, 1018–1028.

[3] Alexander Bondarenko, Maik Fröbe, Meriem Beloucif, Lukas Gienapp, Yamen Ajjour, Alexander Panchenko, Chris Biemann, Benno Stein, Henning Wachsmuth, Martin Potthast, and Matthias Hagen. 2020. Overview of Touché 2020: Argument Retrieval: Extended Abstract. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction: 11th International Conference of the CLEF Association, CLEF 2020, Thessaloniki, Greece, September 22–25, 2020, Proceedings* (Thessaloniki, Greece). Springer-Verlag, Berlin, Heidelberg, 384–395. https://doi.org/10.1007/978-3-030-58219-7_26

[4] Vera Boteva, Demian Gholipour, Artem Sokolov, and Stefan Riezler. 2016. A Full-Text Learning to Rank Dataset for Medical Information Retrieval. *Proceedings of the 38th European Conference on Information Retrieval.* http://www.cl.uni-heidelberg.de/~riezler/publications/papers/ECIR2016.pdf

[5] Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S. Weld. 2020. SPECTER: Document-level Representation Learning using Citation-informed Transformers. In *ACL*.

[6] Pedro Cuenca and Lewis Tunstall. 2024. *Embedding Quantization: Compress Your Embeddings for Fun and Profit.* Hugging Face. https://huggingface.co/blog/embedding-quantization

[7] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2024. LLM.int8(): 8-bit matrix multiplication for transformers at scale. In *Proceedings of the 36th International Conference on Neural Information Processing Systems* (New Orleans, LA, USA) *(NIPS '22)*. Curran Associates Inc., Red Hook, NY, USA, Article 2198, 15 pages.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. https://doi.org/10.18653/v1/N19-1423

[9] Thomas Diggelmann, Jordan Boyd-Graber, Jannis Bulian, Massimiliano Ciaramita, and Markus Leippold. 2020. CLIMATE-FEVER: A Dataset for Verification of Real-World Climate Claims. arXiv:2012.00614 [cs.CL]

[10] Zhen Dong, Zhewei Yao, Amir Gholami, Michael Mahoney, and Kurt Keutzer. 2019. HAWQ: Hessian AWare Quantization of Neural Networks With Mixed-Precision . In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE Computer Society, Los Alamitos, CA, USA, 293–302. https://doi.org/10.1109/ICCV.2019.00038

[11] Jonathan Frankle and Michael Carbin. 2019. The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. In *International Conference on Learning Representations*. https://openreview.net/forum?id=rJl-b3RcF7

[12] Gene H Golub and Christian Reinsch. 1970. Singular value decomposition and least squares solutions. *Numerische mathematik* 14, 5 (1970), 403–420. https://doi.org/10.1007/BF02163027

[13] Song Han, Huizi Mao, and William J. Dally. 2016. Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding. (2016). https://api.semanticscholar.org/CorpusID:2134321

[14] Faegheh Hasibi, Fedor Nikolaev, Chenyan Xiong, Krisztian Balog, Svein Erik Bratsberg, Alexander Kotov, and Jamie Callan. 2017. DBpedia-Entity v2: A Test Collection for Entity Search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Shinjuku, Tokyo, Japan) *(SIGIR '17)*. Association for Computing Machinery, New York, NY, USA, 1265–1268. https://doi.org/10.1145/3077136.3080751

[15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. http://arxiv.org/abs/1503.02531 cite arxiv:1503.02531Comment: NIPS 2014 Deep Learning Workshop.

[16] Geoffrey E Hinton and Ruslan R Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *Science* 313, 5786 (2006), 504–507. https://doi.org/10.1126/science.1127647

[17] Doris Hoogeveen, Karin M. Verspoor, and Timothy Baldwin. 2015. CQADupStack: A Benchmark Data Set for Community Question-Answering Research. In *Proceedings of the 20th Australasian Document Computing Symposium (ADCS)* (Parramatta, NSW, Australia) *(ADCS '15)*. ACM, New York, NY, USA, Article 3, 8 pages. https://doi.org/10.1145/2838931.2838934

[18] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-Efficient Transfer Learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 2790–2799. https://proceedings.mlr.press/v97/houlsby19a.html

[19] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. 2017. Quantized neural networks: training neural networks with low precision weights and activations. *J. Mach. Learn. Res.* 18, 1 (Jan. 2017), 6869–6898.

[20] Shankar Iyer, Nikhil Dandekar, and Kornél Csernai. 2017. First Quora Dataset Release: Question Pairs. In *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, Perth, Australia.

[21] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. 2018. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2704–2713. https://doi.org/10.1109/CVPR.2018.00286

[22] Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. 2014. Speeding up Convolutional Neural Networks with Low Rank Expansions. In *British Machine Vision Conference*.

[23] Herve Jegou, Matthijs Douze, and Cordelia Schmid. 2011. Product Quantization for Nearest Neighbor Search. *IEEE Trans. Pattern Anal. Mach. Intell.* 33, 1 (Jan. 2011), 117–128. https://doi.org/10.1109/TPAMI.2010.57

[24] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for Natural Language Understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, Online, 4163–4174. https://doi.org/10.18653/v1/2020.findings-emnlp.372

[25] Ian T Jolliffe and Jorge Cadima. 2016. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A* 374, 2065 (2016), 20150202. https://doi.org/10.1098/rsta.2015.0202

[26] Jon Marius Kristoffersen. 2024. *Matryoshka Binary Vectors Slash Vector Search Costs with Vespa.* Vespa.ai. https://medium.com/vespa/matryoshka-binary-vectors-slash-vector-search-costs-with-vespa-517019d792d6

[27] Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, and Ali Farhadi. 2022. Matryoshka Representation Learning. In *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (Eds.), Vol. 35. Curran Associates, Inc., 30233–30249. https://proceedings.neurips.cc/paper_files/paper/2022/file/c32319f4868da7613d78af9993100e42-Paper-Conference.pdf

[28] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics* 7 (2019), 452–466. https://doi.org/10.1162/tacl_a_00276

[29] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. 2017. Pruning Filters for Efficient ConvNets. In *International Conference on Learning Representations*. https://openreview.net/forum?id=rJqFGTslg

[30] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Ro{BERT}a: A Robustly Optimized {BERT} Pretraining Approach. https://openreview.net/forum?id=SyxS0T4tvS

[31] Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*. https://openreview.net/forum?id=Bkg6RiCqY7

[32] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. 2018. UMAP: Uniform Manifold Approximation and Projection. *Journal of Open Source Software* 3, 29 (2018), 861. https://doi.org/10.21105/joss.00861

[33] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems*, C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger (Eds.), Vol. 26. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf

[34] Asit Mishra and Debbie Marr. 2018. Apprentice: Using Knowledge Distillation Techniques To Improve Low-Precision Network Accuracy. In *International Conference on Learning Representations*. https://openreview.net/forum?id=B1ae1lZRb

[35] Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2022. MTEB: Massive Text Embedding Benchmark. *arXiv preprint arXiv:2210.07316* (2022). https://doi.org/10.48550/ARXIV.2210.07316

[36] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. *CoRR* abs/1611.09268 (2016). arXiv:1611.09268 http://arxiv.org/abs/1611.09268

[37] Zach Nussbaum, John X. Morris, Brandon Duderstadt, and Andriy Mulyar. 2024. Nomic Embed: Training a Reproducible Long Context Text Embedder. arXiv:2402.01613 [cs.CL]

[38] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Alessandro Moschitti, Bo Pang, and Walter Daelemans (Eds.). Association for Computational Linguistics, Doha, Qatar, 1532–1543. https://doi.org/10.3115/v1/D14-1162

[39] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. http://arxiv.org/abs/1908.10084

[40] Kirk Roberts, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, Kyle Lo, Ian Soboroff, Ellen M. Voorhees, Lucy Lu Wang, and William R. Hersh. 2021. Searching for Scientific Evidence in a Pandemic: An Overview of TREC-COVID. *CoRR* abs/2104.09632 (2021). arXiv:2104.09632 https://arxiv.org/abs/2104.09632

[41] Tara N. Sainath, Brian Kingsbury, Vikas Sindhwani, Ebru Arisoy, and Bhuvana Ramabhadran. 2013. Low-rank matrix factorization for Deep Neural Network training with high-dimensional output targets. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. 6655–6659. https://doi.org/10.1109/ICASSP.2013.6638949

[42] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108* (2019).

[43] Dinghan Shen, Qinliang Su, Paidamoyo Chapfuwa, Wenlin Wang, Guoyin Wang, Ricardo Henao, and Lawrence Carin. 2018. NASH: Toward End-to-End Neural Architecture for Generative Semantic Hashing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Iryna Gurevych and Yusuke Miyao (Eds.). Association for Computational Linguistics, Melbourne, Australia, 2041–2050. https://doi.org/10.18653/v1/P18-1190