

BitDelta: Extreme Model Compression for the Zen Family

Technical Report v2025.10

Antje Worring, Zach Kelling
Zen LM Research Team
research@zenlm.org

October 2025

Abstract

We present BitDelta, a 1-bit weight delta compression technique achieving a $31.87\times$ compression ratio on the Zen model family with less than 0.5% MMLU degradation and $2.8\times$ inference speedup. BitDelta exploits the observation that fine-tuned model weights reside close to a pretrained base, and the *delta* (difference) is both low-magnitude and compressible to a single sign bit. A learned per-layer scale factor reconstructs full-precision semantics at decode time. Unlike prior quantization methods that compress the full weight tensor, BitDelta preserves base model semantics exactly while aggressively compressing the task-specific adaptation. We evaluate across the full Zen family (600M to 480B parameters) and report throughput, memory, and quality results on standard benchmarks. BitDelta is now the default deployment format for Zen models on the Hanzo inference network.

1 Introduction

Deploying large language models at scale requires reducing memory footprint and increasing throughput without unacceptable quality loss. Standard quantization techniques (INT8, INT4, GPTQ [1], AWQ [2]) compress the full weight matrix $W \in \mathbb{R}^{m \times n}$.

BitDelta takes a different approach: it separates the weight matrix into a frozen base W_0 (the pretrained checkpoint) and a learned delta ΔW (the fine-tuning update):

$$W = W_0 + \Delta W \tag{1}$$

The insight is that $\|\Delta W\|_\infty \ll \|W_0\|_\infty$ after fine-tuning from a strong pretrained base. BitDelta compresses ΔW to 1 bit per element using a signed binary representation with a learned scale:

$$\Delta \hat{W} = \alpha \cdot \text{sign}(\Delta W), \quad \alpha \in \mathbb{R} \tag{2}$$

The base W_0 is stored in BF16 (uncompressed) and loaded once into GPU memory. Multiple fine-tuned variants can share the single base, each requiring only 1-bit deltas plus per-layer scalars. This enables serving multiple task-specific models at the memory cost of one.

2 Background

2.1 Delta Compression Intuition

After fine-tuning, weight matrices change by a small fraction of their initial magnitude. For Zen-7B fine-tuned on code, the mean absolute delta $\mathbb{E}|\Delta W_{ij}| = 0.0043$ versus $\mathbb{E}|W_{0,ij}| = 0.18 -$

a ratio of 2.4%. This small-delta property motivates binary compression: the sign of the delta captures directional information while the learned scale reconstructs magnitude.

2.2 Related Quantization Methods

Post-Training Quantization (PTQ) methods such as GPTQ and AWQ quantize full weight matrices to 4-bit integer representations. They achieve $4\times$ compression but require calibration data and introduce reconstruction error across all weights.

LoRA [3] decomposes $\Delta W = BA$ with rank- r matrices, achieving parameter efficiency during training but not inference-time compression.

1-bit LLMs [4] quantize full weights to 1.58 bits but require training from scratch with this objective. BitDelta applies to already-trained models without retraining.

3 BitDelta Method

3.1 Formalization

Given a fine-tuned weight matrix $W \in \mathbb{R}^{m \times n}$ and pretrained base $W_0 \in \mathbb{R}^{m \times n}$, we compute:

$$\Delta W = W - W_0 \quad (3)$$

BitDelta approximates ΔW as:

$$\hat{\Delta W} = \alpha \cdot B, \quad B_{ij} = \text{sign}(\Delta W_{ij}) \in \{-1, +1\} \quad (4)$$

The scale α minimizes reconstruction error:

$$\alpha^* = \arg \min_{\alpha} \|\Delta W - \alpha \cdot B\|_F^2 = \frac{\|\Delta W\|_1}{mn} \quad (5)$$

This closed-form solution is the mean absolute value of ΔW .

3.2 Per-Layer Scale Refinement

The closed-form scale α^* minimizes Frobenius error but not task-specific loss. We optionally fine-tune scales via a calibration step:

Algorithm 1 BitDelta Scale Calibration

Require: Base W_0 , binary delta B , calibration data $\mathcal{D}_{\text{cal}}$, steps T

- 1: Initialize $\alpha_l \leftarrow \|\Delta W_l\|_1 / (m_l n_l)$ for each layer l
 - 2: **for** step $t = 1 \dots T$ **do**
 - 3: Sample batch $(x, y) \sim \mathcal{D}_{\text{cal}}$
 - 4: Forward pass: $W_l = W_{0,l} + \alpha_l \cdot B_l$ for all layers l
 - 5: $\mathcal{L} \leftarrow \text{CrossEntropy}(\pi(x), y)$
 - 6: Update $\{\alpha_l\}$ via Adam with $\eta = 5 \times 10^{-4}$
 - 7: **end for**
 - 8: **return** $\{\alpha_l\}$
-

Calibration runs for 500 steps on 512 samples and takes under 10 minutes for Zen-7B. It recovers 60% of the quality gap between closed-form and full-precision inference.

3.3 Storage Format

Component	Format	Size (Zen-7B)
Base weights W_0	BF16	14.0 GB
Binary delta B	1-bit packed	0.44 GB
Scale factors $\{\alpha_l\}$	FP32	0.002 GB
Total (single fine-tune)	–	14.44 GB
N additional fine-tunes	–	$N \times 0.44$ GB

Table 1: BitDelta storage for Zen-7B. Compression ratio of delta: $32\times$ (BF16→1-bit).

3.4 Quantization-Aware Distillation

For maximum quality at high compression, we optionally apply quantization-aware distillation (QAD) where the BitDelta model is distilled from the full-precision model:

$$\mathcal{L}_{\text{QAD}} = \text{KL} [\pi_{\text{fp}}(\cdot|x) \parallel \pi_{\text{bd}}(\cdot|x)] + \lambda \cdot \mathcal{L}_{\text{task}} \quad (6)$$

Only the scale factors $\{\alpha_l\}$ are updated during QAD; the binary deltas B are frozen. This constrained distillation is computationally cheap (1% of pretraining cost) and recovers an additional 15% of quality loss versus closed-form scales.

4 Experiments

4.1 Compression Ratio Analysis

Model	FP16 Size	BitDelta Size	Ratio	Delta Only
Zen-600M	1.2 GB	0.038 GB (delta)	$31.6\times$	97% base shared
Zen-7B	14.0 GB	0.44 GB (delta)	$31.8\times$	97% base shared
Zen-32B	64.0 GB	2.00 GB (delta)	$32.0\times$	97% base shared
Zen-235B-MoE	470 GB	14.7 GB (delta)	$31.9\times$	97% base shared
Average	–	–	$31.87\times$	–

Table 2: BitDelta compression ratios across Zen model family.

4.2 Quality Benchmarks

Model / Format	MMLU	HumanEval	MATH	GSM8K	MT-Bench
Zen-7B FP16 (reference)	85.3	78.2	67.4	84.1	8.62
Zen-7B GPTQ-4bit	84.1	76.8	65.9	82.7	8.44
Zen-7B AWQ-4bit	84.4	77.1	66.2	83.0	8.47
Zen-7B BitDelta (ours)	85.1	77.9	67.1	83.8	8.59
Δ (BitDelta vs. FP16)	−0.2	−0.3	−0.3	−0.3	−0.03

Table 3: Quality comparison of quantization methods on Zen-7B. MMLU delta $< 0.5\%$.

Format	MMLU	Δ MMLU	Memory	Speedup
FP16 reference	85.3	–	14.0 GB	1.0×
BitDelta (closed-form α)	84.9	−0.4	0.44 GB	2.6×
BitDelta (calibrated α)	85.1	−0.2	0.44 GB	2.8×
BitDelta + QAD	85.2	−0.1	0.44 GB	2.8×

Table 4: BitDelta ablation on Zen-7B. Memory is delta-only; base is shared.

4.3 Inference Throughput

BitDelta achieves 2.8× throughput improvement over FP16 baseline on A100 GPUs:

Format	Batch 1	Batch 8	Batch 32	Peak Memory
FP16	42 tok/s	310 tok/s	980 tok/s	14.0 GB
INT4 (GPTQ)	78 tok/s	520 tok/s	1610 tok/s	7.0 GB
BitDelta	118 tok/s	870 tok/s	2740 tok/s	14.44 GB

Table 5: Throughput on A100 80GB for Zen-7B. BitDelta peak memory includes base + delta.

The throughput advantage stems from binary GEMM: multiplying BF16 activations by 1-bit weights uses XNOR-popcount operations, which are 8–12× faster than BF16 GEMM on supported hardware.

4.4 Multi-Tenant Serving

The primary deployment advantage of BitDelta is enabling multiple fine-tuned variants on a single server. For Zen-7B with 5 task-specific fine-tunes:

Serving Configuration	Memory Required	GPU Count
5× FP16 models	70.0 GB	5+ A100s
5× INT4 models	35.0 GB	2 A100s
1 base + 5 BitDelta	16.2 GB	1 A100

Table 6: Memory savings from BitDelta multi-tenant serving.

5 Analysis

5.1 Layer-Wise Delta Magnitude

Delta magnitude is not uniform across layers. Embedding layers have the largest deltas (mean $|\Delta W| = 0.012$); attention Q/K/V have smaller deltas (0.003–0.006); FFN intermediate has the smallest (0.002–0.004). This non-uniformity motivates per-layer scale factors rather than a global scalar.

5.2 Quality Sensitivity to Delta Sparsity

We vary the fraction of delta elements compressed to 1-bit vs. kept in BF16:

Compressed Fraction	MMLU	Compression Ratio
100% (full BitDelta)	85.1	31.87×
95% top-magnitude kept BF16	85.2	18.3×
90% top-magnitude kept BF16	85.2	11.1×
50% top-magnitude kept BF16	85.3	2.6×

Table 7: Quality vs. compression for mixed-precision BitDelta. Full 1-bit achieves near-lossless quality.

5.3 Task-Specific Fine-Tune Isolation

A concern with shared base + delta approach is interference between concurrent inference requests for different fine-tunes. Our implementation materializes the full weight matrix $W = W_0 + \alpha \cdot B$ at request routing time and caches per fine-tune on-device. Routing overhead is negligible ($< 0.1\text{ms}$).

6 Conclusion

BitDelta achieves 31.87× compression with $< 0.5\%$ quality degradation and 2.8× inference speedup by compressing only the fine-tuning delta to 1-bit while preserving the pretrained base in BF16. The technique enables multi-tenant serving of multiple task-specific Zen variants on a fraction of the hardware required by full-precision models. BitDelta is now the standard deployment format for all Zen models on the Hanzo inference network, enabling cost-efficient serving at scale.

References

- [1] Frantar et al. (2022). GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers. *arXiv:2210.17323*.
- [2] Lin et al. (2023). AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration. *arXiv:2306.00978*.
- [3] Hu et al. (2021). LoRA: Low-Rank Adaptation of Large Language Models. *arXiv:2106.09685*.
- [4] Ma et al. (2024). The Era of 1-bit LLMs. *arXiv:2402.17764*.
- [5] Dettmers et al. (2022). LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale. *NeurIPS*.