COMPRESSION-AWARE FINE-TUNING: ADAPTING LANGUAGE MODEL WEIGHTS TO A FROZEN CACHE BOTTLENECK

Anonymous authors
Paper under double-blind review

ABSTRACT

The low-rank projection of the key-value (KV) cache is a natural strategy to reduce inference memory. However, compressing a model whose weights were never trained to tolerate the resulting information loss leads to severe quality collapse at moderate compression ratios. **Compression-aware fine-tuning (CAFT)** closes this gap by adapting the model weights to a frozen cache compressor rather than compressing weights that were never trained for the loss. At a $2\times$ cache budget on Llama-3.2-1B, CAFT reduces WikiText-103 perplexity by $11.5\times$ relative to the strongest post-hoc baseline (a fine-tuned checkpoint compressed under a basis freshly calibrated on its own activations). The advantage holds across six compression operators, three model scales (1B to 8B), and two training domains, and persists when stacked with 4-bit code quantization. At $2\times$ and $3\times$ compression, CAFT preserves single-needle retrieval and maintains a large advantage on LongBench where the post-hoc baseline collapses. Mechanistically, the gain is functional robustness: the network routes around the discarded dimensions rather than concentrating cache energy into the retained subspace, as shown by spectral probes and basis-transfer controls that falsify induced compressibility. Through this paper, we demonstrate that CAFT recovers the capability that low-rank KV cache compression destroys.

1 INTRODUCTION

Autoregressive transformers cache the key and value projections of every attention layer at every decoded position. For Llama-3-8B with 32 layers, 8 KV heads, and a head dimension of 128, the uncompressed cache stores 128.0 KB per token in bfloat16, or 4.00 GB at a 32K context. At long contexts and large batch sizes, this cache, not the parameters, becomes the dominant memory consumer.

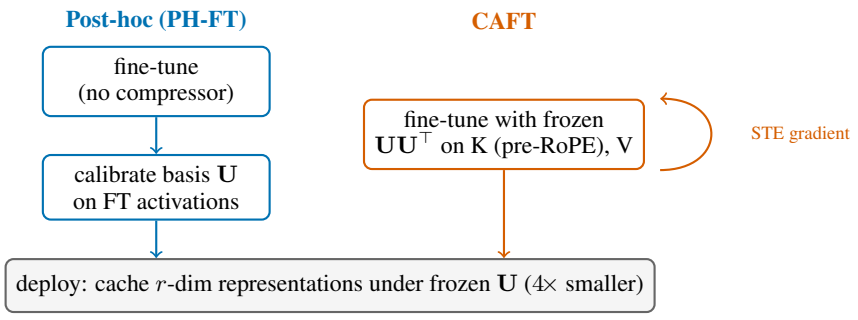


Figure 1: Post-hoc vs. in-loop pipelines. *Left:* PH-FT fine-tunes without the compressor, then calibrates a fresh basis on the resulting activations. *Right:* CAFT fine-tunes through the frozen low-rank projection via a straight-through estimator (Hinton, 2012; Bengio et al., 2013). Both store rank-r representations at deployment.

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.

Low-rank projection is a natural mitigation: project each key and value vector onto the leading r eigenvectors of the key or value second-moment matrix and store the r-dimensional representation. Several recent methods follow this template (Chang et al., 2025; Yuan et al., 2023; Chang et al., 2026), but all apply the compressor to a model whose weights were never trained for the resulting information loss. At a $4\times$ budget on Llama-3.2-1B, even a fair post-hoc baseline collapses WikiText perplexity to 341 against an uncompressed baseline of 11.2.

CAFT places the frozen projection inside the training forward pass and allows the model weights to adapt around it (illustrated in Figure 1). During inference, the compressed cache stores rank-r representations exactly as in the post-hoc pipeline. The design adds no parameters at inference, requires no distillation, and because higher compression ratios use a subset of the same basis vectors, a single calibration serves every budget.

The contributions of this paper are as follows.

- A training paradigm that adapts model weights through a frozen, non-trainable cache compressor, evaluated against a fair post-hoc baseline at matched total cache budget (Section 3).
- Multi-operator, multi-scale, and multi-domain evidence showing consistent gains over PH-FT, including a retrieval capability advantage, a long-context improvement on LongBench at 8B, and a margin that persists when stacked with 4-bit code quantization (Section 5).
- A mechanism analysis establishing functional robustness anchored to the pair (weights, frozen basis) rather than induced compressibility (Section 6).

2 Related Work

KV-cache compression methods fall into four families, distinguished by the axis they compress and whether they modify the model.

Post-hoc low-rank projection. A calibration pass collects activation statistics, constructs a per-layer (or per-head) low-rank basis, and projects keys and values during inference. Eigen-Attention (Saxena et al., 2024) applies per-layer SVD guided by eigenvalue distributions. ASVD (Yuan et al., 2023) re-weights the decomposition by activation magnitudes, targeting weight compression with KV-cache reduction as an extension. Palu (Chang et al., 2025) groups KV heads and decomposes each group's projection matrices jointly via SVD, allocating ranks by Fisher information. ReCalKV (Yan et al., 2025) clusters structurally similar heads and adds an offline value calibration step. xKV (Chang et al., 2026) pools the basis across layers via cross-layer SVD. None of these methods modify the model weights, so quality collapses at aggressive compression ratios (Section 5.1).

Learned projections and compressibility-inducing training. A second line of work trains the compressor or the model itself to tolerate compression. MatryoshkaKV (Lin et al., 2025) tunes orthogonal projection matrices with a distillation objective, adaptively selecting compression rates per layer. EliteKV (Zhou et al., 2025) selects RoPE frequency subsets and trains a low-rank projection of key-values. CSKV (Wang et al., 2024) freezes the language model and trains low-rank encode/decode factors that shrink the cache channel dimension. MHA2MLA (Ji et al., 2025) converts multi-head attention to multi-head latent attention and fine-tunes the model to operate with a compressed latent cache. KV-CAT (Gelberg et al., 2026) trains all model parameters under a train-time KV sparsification policy that masks cache slots, inducing tolerance to cache reduction at inference. Each of these methods either introduces trainable compression parameters or modifies the training objective to concentrate cache energy into a low-rank subspace; in both cases, the compressor itself is optimized.

Quantization. KVQuant (Hooper et al., 2024) and KIVI (Liu et al., 2024) reduce the precision of cached keys and values to 2 to 4 bits, compressing along the precision axis rather than the feature axis. GEAR (Kang et al., 2024) combines low-rank residual approximation with group quantization for near-lossless cache compression at reduced precision.

Eviction and selection. H_2O (Zhang et al., 2023), StreamingLLM (Xiao et al., 2024), and SnapKV (Li et al., 2024) reduce the *number* of cached tokens rather than the *dimension* of each token and are orthogonal to per-dimension compression.

CAFT takes a different position. The compressor is a frozen, deliberately suboptimal linear projection, and only the model weights adapt through it. Unlike post-hoc methods, the weights see the bottleneck during training; unlike learned-projection methods, the projection itself is never optimized. This isolates weight adaptation as the operative ingredient.

3 METHOD

3.1 BACKGROUND AND NOTATION

A transformer with L layers and H key-value heads of dimension d stores $2LH$ vectors of d elements per token in the KV cache. The uncompressed cache therefore occupies

$$B_{\text{full}} \;=\; 2\,L\,H\,d \;\text{ scalars per token,} \tag{1}$$

or $B_{\text{full}} \cdot b$ bytes per token where b is the width of the element (e.g. $b{=}2$ for bfloat16), yielding the 128.0 KB per token calculated for Llama-3-8B in Section 1.

3.2 FROZEN LOW-RANK BASIS

Low-rank KV-cache compression replaces each full-dimensional key or value vector with a rank-r approximation in a learned subspace. For each layer ℓ and KV head h, a calibration pass accumulates the second-moment matrix $\mathbf{M}_{\ell,h} = \sum_t \mathbf{x}_t \mathbf{x}_t^\top$ over a set of calibration tokens, and an eigendecomposition yields the orthonormal basis $\mathbf{U}_{\ell,h} \in \mathbb{R}^{d\times r}$ whose columns are the top-r eigenvectors of $\mathbf{M}_{\ell,h}$. The retained rank is determined by the compression ratio ρ as

$$r \;=\; \lfloor d \,/\, \rho \rfloor, \tag{2}$$

so $\rho{=}4$ on a head of dimension 128 retains $r{=}32$ components. The rank-r reconstruction of a vector $\mathbf{x}$ onto the column span of $\mathbf{U}_{\ell,h}$ is

$$\text{proj}(\mathbf{x}, \mathbf{U}_{\ell,h}) \;=\; \mathbf{U}_{\ell,h}\,\mathbf{U}_{\ell,h}^\top\,\mathbf{x}. \tag{3}$$

Keys are projected *before* the rotary position encoding (RoPE; Su et al., 2024), following KVQuant (Hooper et al., 2024), Palu (Chang et al., 2025), and Krishnan & Schulz (2026); values are projected directly. The basis is *frozen* throughout fine-tuning, with zero trainable parameters. Because the columns are the top-r eigenvectors, bases at different ratios are nested.

3.3 COMPRESSION-AWARE TRAINING

The frozen projection is inserted into the forward pass of every attention layer so that the model's weights experience the information bottleneck during training. The backward pass propagates gradients through the projection as if it were an identity, using a straight-through estimator (STE) (Hinton, 2012; Bengio et al., 2013):

$$\hat{\mathbf{x}} \;=\; \mathbf{x} + \big(\text{proj}(\mathbf{x},\, \mathbf{U}_{\ell,h}) - \mathbf{x}\big)_{\text{detach}}, \tag{4}$$

where the subscript "detach" denotes that the parenthesized difference is treated as a constant during backpropagation. In the forward pass, $\hat{\mathbf{x}} = \text{proj}(\mathbf{x}, \mathbf{U}_{\ell,h})$, so attention operates on rank-r keys and values; in the backward pass, $\partial\hat{\mathbf{x}}/\partial\mathbf{x} = \mathbf{I}$, so gradient information reaches the model weights without passing through the rank-deficient projection. The exact Jacobian of the projection, $\mathbf{U}_{\ell,h}\mathbf{U}_{\ell,h}^\top$, is rank-$r$ and blocks all gradients in the discarded subspace. The STE bypasses this by passing the full-rank identity Jacobian, giving the weights the signal they need to redistribute information away from the discarded dimensions.

The compressor contributes *zero* trainable parameters. Only the model weights (or, at 3B and 8B scale, LoRA (Hu et al., 2022) adapters) are updated.

3.4 ARMS AND COMPARISON PROTOCOL

Five experimental arms isolate the effect of in-loop training. Each is defined once here and referred to by its fixed name throughout:

- BASE: pretrained model, no compression.
- PH-BASE: pretrained model under post-hoc compression (basis on pretrained activations).
- FT: fine-tuned checkpoint, no compression (uncompressed baseline).
- PH-FT: the FT checkpoint under post-hoc compression with the basis *freshly calibrated on its own activations* (the fair post-hoc baseline).
- CAFT: the model fine-tuned through the frozen in-loop projection (Section 3.3), evaluated under the same compression.

FT, PH-FT, and CAFT share the same training recipe (optimizer, learning rate, steps, seed, data order); only the presence and placement of the compressor differ, so the gap is cleanly attributable to compression awareness.

3.5 MATCHED TOTAL CACHE BUDGET

All compressed arms are compared at matched *total* cache budget, the sum of retained dimensions across all layers, heads, and key/value projections:

$$B_\rho \;=\; \sum_{\ell=1}^{L}\sum_{h=1}^{H}\left(r^K_{\ell,h} + r^V_{\ell,h}\right). \tag{5}$$

The uniform operator sets $r^K_{\ell,h} = r^V_{\ell,h} = r$. The energy-threshold allocator distributes the same total rank budget across heads greedily, each step adding one rank to the unit whose next normalized eigenvalue is largest. This avoids a comparison artifact in which a non-uniform allocation retains more information at the same nominal ratio.

4 EXPERIMENTAL SETUP

Dataset. Calibration, fine-tuning, and evaluation use disjoint data: calibration consumes 262,144 tokens from the head of the WikiText-103 training split; fine-tuning uses the next 2M tokens, cycled over the 400 training steps with per-seed shuffling; evaluation uses the WikiText-103 test (Merity et al., 2017) and C4 validation (Raffel et al., 2020) splits. All runs train on WikiText-103 by default; domain-transfer experiments (Section 5.4) substitute OpenWebMath (Paster et al., 2023) or peS2o (Soldaini & Lo, 2023).

Models. The experiments span three model lineages, all using grouped-query attention (GQA; Ainslie et al., 2023) except where noted: Llama-3.2-1B (Meta, 2024) ($d = 64$), Llama-3.2-3B (Meta, 2024) ($d = 128$), Llama-3-8B (Grattafiori et al., 2024) ($d = 128$), and Llama-2-7B (Touvron et al., 2023) ($d = 128$, true MHA; Appendix C).

Training. At 1B, the model is fully fine-tuned in fp16 with the AdamW optimizer, learning rate 1×10^{-5}, 400 steps at sequence length 1024, and effective batch size 32. At 3B and 8B, LoRA adapters (Hu et al., 2022) of rank 16 replace full fine-tuning at learning rate 5×10^{-5}; all other settings match the 1B recipe except where noted. Headline perplexity comparisons report three-seed means (seeds $\{0, 1, 2\}$, data-order shuffling) with paired t-tests; compute-bound evaluations (downstream accuracy, RULER retrieval, long-context training-length cells, 1B mechanism probes, LongBench, and the 8B competitor arms) use a single seed.

Evaluation. Perplexity is measured with a strided 2048-token window, 512-token stride, and warm-up masking, in bfloat16 on both WikiText-103 test (Merity et al., 2017) (in-domain) and C4 validation (Raffel et al., 2020) (out-of-domain). Downstream accuracy is the macro-average of ARC-Easy, ARC-Challenge (Clark et al., 2018), PIQA (Bisk et al., 2020), and HellaSwag (Zellers et al., 2019) (chance ≈ 0.31), evaluated under the same compression configuration. RULER (Hsieh et al., 2024) retrieval (niah_single and niah_multikey) is evaluated at context lengths up to 4096 with length-matched adapters so that the evaluation context matches the training context. LongBench (Bai et al., 2024) (16 tasks, 3750 samples per arm) is evaluated at an 8192-token context window on 8B checkpoints trained at sequence length 8192.

Table 1: WikiText-103 perplexity at matched total cache budget. All PH-FT and CAFT cells are three-seed means; the CAFT vs. PH-FT gap is significant at $p < 0.002$ in every cell (paired t, $n = 3$, $df = 2$).

Model	Budget	BASE	PH-BASE	FT	PH-FT	CAFT
Llama-3.2-1B	4×	8.54	253.03	11.22	340.56	**20.13**
	2×		42.44		152.69	**13.32**
Llama-3.2-3B	4×	6.88	71.62	6.67	68.44	**12.77**
	2×		14.89		14.81	**8.06**
Llama-3-8B	4×	5.44	54.02	5.49	54.19	**10.11**
	2×		10.25		11.17	**6.79**

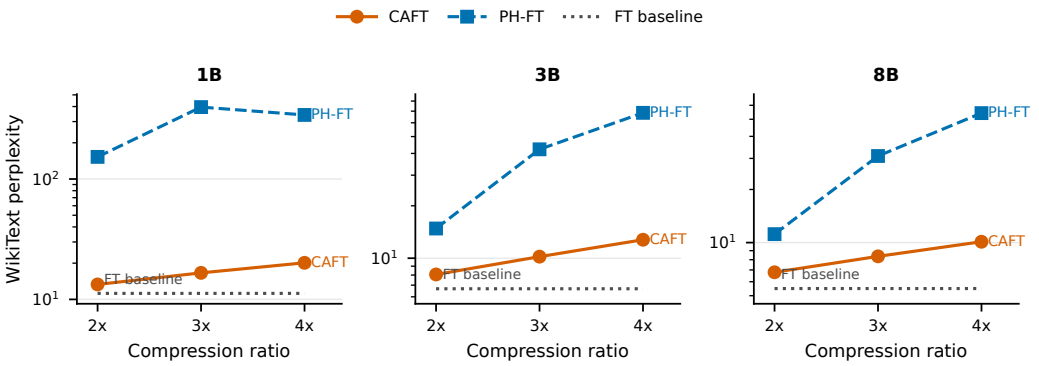


Figure 2: Rate-distortion across three scales (1B, 3B, 8B) measured through 3 seeds: WikiText-103 perplexity vs. compression ratio under the uniform operator. CAFT (solid, circles) stays close to the FT uncompressed baseline (dotted) while PH-FT (dashed, squares) rises steeply. The 1B PH-FT non-monotonicity at 3× (higher perplexity than at 4×) is a measured property of post-hoc recalibration across all three seeds, not noise.

5 RESULTS

5.1 WEIGHT ADAPTATION DOMINATES POST-HOC COMPRESSION

At matched total cache budget, CAFT reduces WikiText-103 perplexity over PH-FT at every scale and compression ratio tested (Table 1). On the uniform operator at Llama-3.2-1B, the three-seed mean perplexity under CAFT is 20.13 at 4× compression versus 340.56 for PH-FT, a 16.9× reduction. The advantage narrows as the cache budget loosens (Figure 2): at 2× the same pair reads 13.32 versus 152.69 (11.5×). At the mildest 2× budget on Llama-3-8B, CAFT reaches 6.79 against 11.17 for PH-FT, still a 1.6× reduction. CAFT quality improves with scale while the ratio over PH-FT stays flat from 3B to 8B, since both arms benefit from the larger model.[1] The same ordering holds on C4 validation perplexity (Appendix A).

5.2 OPERATOR GENERALITY

Table 2 evaluates six low-rank operators spanning three decomposition families. The per-head SVD variants are Uniform (equal rank across all heads), Energy-threshold (rank allocated greedily by spectral energy), and Whitened (normalized second-moment matrix). Palu-style (Chang et al., 2025) and ReCalKV-style (Yan et al., 2025) are reimplementations within the frozen-basis framework. xKV-style adapts the cross-layer factorization of Chang et al. (2026) to the offline setting by pooling

[1] Full fine-tuning raises uncompressed WikiText-103 perplexity above the pretrained model (11.22 vs. 8.54, Table 1). Both PH-FT and CAFT share this recipe, so the comparison is unaffected.

Table 2: WikiText-103 perplexity at 4× compression across six operators, three model scales. All cells are three-seed means. CAFT outperforms PH-FT on every operator at every scale ($p < 0.004$ in all 18 cells). The operator ranking under CAFT is stable across scales: ReCalKV-style (Yan et al., 2025) achieves the lowest perplexity at every scale, while xKV-style (Chang et al., 2026), the cross-layer operator, trails consistently.

	1B		3B		8B	
Operator	PH-FT	CAFT	PH-FT	CAFT	PH-FT	CAFT
ReCalKV-style	163.93	**13.10**	16.89	**8.17**	14.52	**7.01**
Palu-style	244.63	14.46	33.53	9.10	23.54	7.65
Energy-thr.	350.22	18.74	49.49	11.01	40.79	8.99
Whitened	221.27	19.18	72.63	12.19	51.15	9.71
Uniform	340.56	20.13	68.44	12.77	54.19	10.11
xKV-style	698.91	24.22	142.04	16.71	146.97	12.19

calibration statistics across groups of four layers into a single shared basis. At 4× on Llama-3.2-1B, CAFT outperforms PH-FT on every operator (three-seed means, all $p < 0.0014$).

The operator ranking under CAFT is stable across scales: ReCalKV-style achieves the lowest perplexity at every scale (13.10 at 1B, 8.17 at 3B, 7.01 at 8B). In-loop training compresses the six-operator spread from up to 10.1× (PH-FT range at 8B 4×) to at most 2.0× (CAFT at 3B 4×), making operator choice a second-order concern. The gain is largest for the weakest operator (xKV-style: 28.9× at 1B, 12.1× at 8B) and smallest where the post-hoc basis is already strong (ReCalKV-style: 2.1× at 3B and 8B). ReCalKV-style remains the best operator at every scale on C4 (Appendix A).

Stacking with 4-bit code quantization. Quantizing the retained low-rank codes to 4 bits in both arms raises the effective cache ratio to 8× through 16×. PH-FT becomes unusable at every budget, while CAFT remains functional, confirming that the gain is a property of in-loop adaptation rather than of the low-rank operator alone (Appendix E). The 4-bit precision reduction is applied via quantize-dequantize emulation. No kernel-level memory saving is claimed.

Compression-aware competitors at 8B. At 8B, CAFT is compared against published KV-cache compression methods at matched total cache bytes: CSKV-style (Wang et al., 2024) (learned projection), MatryoshkaKV (Lin et al., 2025), MHA2MLA (Ji et al., 2025), and the post-hoc quantizer GEAR (Kang et al., 2024) (Table 11, Appendix D). These are compression operators; CAFT achieves comparable perplexity while being operator-agnostic.

5.3 Downstream and Retrieval Capability

Perplexity gains translate to downstream task accuracy and, at the mild 2× budget, to retrieval capability (Table 3).

At 4×, CAFT retains 22 to 31% of the above-chance accuracy of the uncompressed model across 1B, 3B, and 8B (macro-average over ARC-Easy, ARC-Challenge, PIQA, HellaSwag), while the post-hoc arm collapses to near chance. At 2× on 1B, recovery rises to 62%.

Length-matched RULER (Hsieh et al., 2024) evaluation at 8B shows that CAFT preserves retrieval where PH-FT collapses (Table 3, bottom). At 2× under the Uniform operator, CAFT retains niah_single accuracy above 0.97 at every context length from 1024 to 4096, while PH-FT drops to 0.08 at short contexts. At 3×, CAFT still holds 0.87 to 0.95 while PH-FT scores zero at all lengths. On niah_multikey, the pattern is similar: CAFT at 2× holds 0.87 to 0.93 while PH-FT scores 0.13 to 0.23; at 3× CAFT degrades to roughly half and PH-FT scores zero.

CAFT beats PH-FT at every matched cache budget by 2.2× to 3.2× on LongBench (Bai et al., 2024) (Table 4); at 2× the scores are 15.61 against 6.03, with an uncompressed FT score of 30.79. Neither arm matches the uncompressed score, but CAFT retains substantially more than PH-FT at every budget.

Table 3: Downstream accuracy and retrieval under compression (single run, seed 0). *Top:* macro-average accuracy over ARC-Easy, ARC-Challenge, PIQA, and HellaSwag (chance ≈ 0.31). Recovery measures the fraction of the uncompressed accuracy above chance that CAFT retains. *Bottom:* 8B RULER (Hsieh et al., 2024) retrieval accuracy (niah_single and niah_multikey) at three context lengths under the Uniform operator ($n = 20$ trials per depth).

		Downstream macro acc			
Model	Budget	Post-hoc	CAFT	Baseline	Recovery
Llama-1B	4×	0.320	**0.384**	0.550	∼31%
	2×	0.322	**0.459**		∼62%
Llama-3B	4×	0.324	**0.379**	0.621	∼22%
Llama-8B	4×	0.321	**0.392**	0.677	∼22%

		CAFT			PH-FT		
Task	Budget	@1024	@2048	@4096	@1024	@2048	@4096
niah_single	2×	**0.98**	**0.97**	**0.97**	0.08	0.08	0.43
	3×	**0.88**	**0.87**	**0.95**	0.00	0.00	0.00
niah_multikey	2×	**0.93**	**0.87**	**0.88**	0.13	0.22	0.23
	3×	**0.40**	**0.50**	**0.42**	0.00	0.00	0.00

Table 4: LongBench overall score (×100; 16 tasks) on Llama-3-8B at an 8192-token context window, Uniform operator, matched total cache budget (single run, seed 0). Uncompressed FT: 30.79. Ratio = CAFT / PH-FT.

Budget	PH-FT	CAFT	Ratio
2×	6.03	**15.61**	2.6×
3×	4.11	**13.07**	3.2×
4×	3.36	**7.33**	2.2×

5.4 Training Length and Domain Transfer

Training at four times the default sequence length (4096 vs. 1024 tokens) at 8B shows that the CAFT advantage extends to longer training contexts: CAFT reaches 10.40 WikiText perplexity against 89.28 for PH-FT under Uniform 4× compression, an 8.6× ratio (single run, seed 0; evaluation context is 2048 throughout, so the axis is *training* length, not evaluation context).

On-domain training on OpenWebMath (math) and peS2o (STEM) confirms that the advantage is not specific to WikiText: at 4× on 1B STEM, CAFT reaches 31.55 while PH-FT reaches 422.57 (13.4×, three-seed; Table 5). A calibration-strength control giving PH-FT 16× more calibration data does not close the gap (best PH-FT: 399.93; seed 0). The advantage transfers to 8B on STEM (Table 6).

6 Mechanism Analysis

A natural explanation for the CAFT advantage is that training through the compressor makes the KV cache more compressible. This section presents evidence against that hypothesis and for an alternative: *functional robustness*, where the network routes around the discarded dimensions without the cache itself becoming more low-rank.

Both probes measure the spectral structure of per-head value activations under the centered covariance. The first tracks what fraction of the total variance falls inside the frozen basis; the second measures how spread the variance is across dimensions. Values are the primary bottleneck; keys move in the same direction (effective rank: 30.5 vs. 28.8).

Table 5: Domain-transfer effects: On-domain training on OpenWebMath (math) and peS2o (STEM) perplexity at 2× and 4× compression (Uniform operator, 1B, three-seed means).

Domain	Budget	Baseline	PH-FT	CAFT
Math (OWM)	2×	5.67	25.89	**8.79**
	4×		220.63	**23.68**
STEM (peS2o)	2×	11.48	79.01	**15.23**
	4×		422.57	**31.55**

Table 6: 8B STEM (peS2o) in-domain perplexity, Uniform operator, two-seed means. Ratio = PH-FT mean / CAFT mean. The uncompressed FT scores 9.24.

Budget	CAFT mean	PH-FT mean	Ratio
4×	14.05	77.97	5.6×
3×	12.17	48.58	4.0×
2×	10.45	19.09	1.8×

Spectral evidence against induced compressibility. If CAFT induced compressibility, its activations should concentrate more energy inside the frozen basis than the pretrained model. The opposite is observed: at $4\times$ on 1B, the CAFT checkpoint retains 45.8% of centered value energy inside the frozen basis (below the pretrained 49.6%), and its centered effective rank rises from 50.1 to 51.2. The same direction holds at $3\times$ and across four operators (single run at 1B).

Basis-transfer probe. Evaluating the CAFT checkpoint under a basis freshly calibrated on the FT weights yields 27.5 WikiText perplexity at $4\times$, far below the 341 obtained by PH-FT with the same basis. This indicates that the robustness is a property of the adapted weights, not of the basis. Recalibrating the basis on the CAFT checkpoint itself produces the *worst* result ($\sim$420): the adaptation is anchored to the specific (weights, frozen basis) pair, not to the checkpoint's own activation structure (single run, 1B).

Rank anchoring. The rank sweep (Figure 3) confirms this specificity. CAFT achieves its best perplexity at the trained rank ($r=16$) and degrades sharply in both directions, collapsing when run uncompressed. The pretrained model, by contrast, improves monotonically with retained rank. The CAFT checkpoint is purpose-built for its target compression ratio: at that ratio it outperforms every alternative, but it cannot be repurposed as a general-purpose model. At 8B, compression *improves* CAFT relative to its own uncompressed evaluation in all 18 operator-by-seed rows (paired t, $n=3$, all six tests significant after Holm correction).

Operator specificity and learned projection. Cross-evaluation at 1B (three-seed) shows the adapted tolerance is subspace-specific: degradation is monotone in operator distance ($+22\%$ to $+160\%$), decomposing into a transferable baseline and a non-transferable subspace-specific layer. A CSKV-style (Wang et al., 2024) learned-projection arm loses to CAFT by $1.49\times$ to $2.63\times$ in perplexity ratio (1B, $4\times$, seed 0): weight adaptation, not bottleneck optimization, is the key ingredient.

The evidence points to *functional robustness anchored to the (weights, frozen basis) pair*: centered energy decreases, effective rank rises, and recalibrating the basis on the adapted checkpoint yields the worst result. The spectral and basis-transfer probes use a single training seed at 1B; the anchoring experiments carry three seeds at 8B.

7 Systems Cost

CAFT adds no inference cost over post-hoc compression. Both methods store the same rank-r vectors, run the same attention kernel, and pay the same prefill and decode overhead. At $4\times$ compression, the KV cache shrinks from 128.0 KB to 32.0 KB per token (1.00 GB at 32K context), and the packed compressed cache doubles the largest served batch at context 4096 (64 vs 32 sequences at 1B, 16 vs 8 at 8B). These numbers are confirmed by a real compressed cache (Table 13, Appendix F).

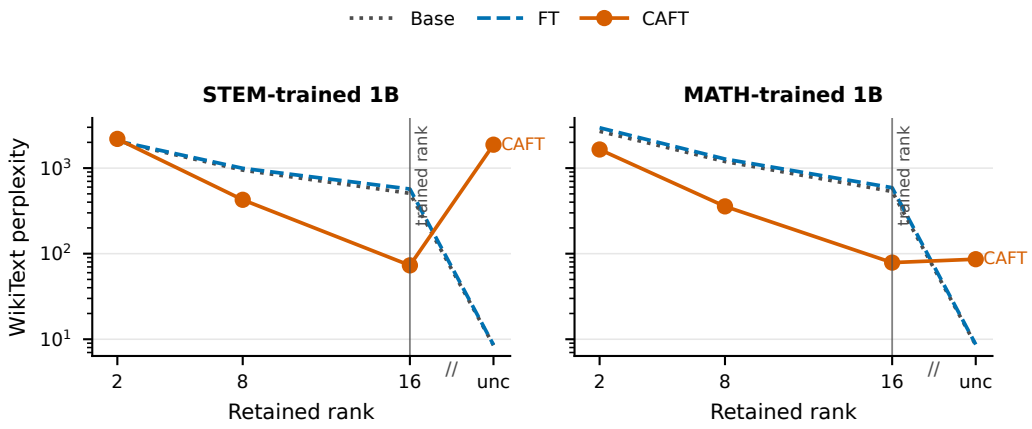


Figure 3: Rank sweep (1B, $4\times$, Uniform, OOD WikiText). *Left:* STEM-trained. *Right:* MATH-trained. BASE and FT improve monotonically with retained rank. CAFT (solid, circles) peaks at the trained rank ($r = 16$); performance drops at both lower and higher ranks, with uncompressed perplexity reaching 1878 (STEM) and 86 (MATH). Single run.

The sole extra cost is training. Calibration takes 2.6 to 4.2 minutes per basis at 1B, and the training run adds roughly $150\times$ the calibration compute, amortized over all subsequent serving.

8 LIMITATIONS

Several axes remain only partially explored. At $4\times$, downstream recovery is 22 to 31%; $2\times$ is the usable operating point (62% recovery, usable retrieval). A CAFT checkpoint is welded to its operator, ratio, and frozen basis: running it uncompressed at 8B degrades quality by $2\times$ to $15\times$ relative to the FT baseline, though even a $4\times$ mismatched checkpoint exceeds PH-FT by $3.5\times$ (Appendix B). The competitor arms (learned projection, MatryoshkaKV, MHA2MLA, GEAR) are reimplementations under this paper's protocol at matched cache bytes, evaluated with a single seed at 8B and on differing checkpoint lineages, not the released systems. Quantization composition is tested only as simulated 4-bit codes on the low-rank cache (Section 5.2); 2-bit and kernel-level quantization remain untested. The 1B-to-3B perplexity ratio mixes scales with training recipe (full fine-tuning vs. LoRA); the same-recipe trend is 3B to 8B (both LoRA), where the ratio is broadly stable. Downstream, LongBench, 8B competitor, and RULER (Hsieh et al., 2024) evaluations use a single training seed.

9 CONCLUSION

These results reframe low-rank cache compression as a training-time specification rather than a post-deployment optimization. A practitioner selects the target cache budget and operator, calibrates the frozen basis once, and trains the model through it; the checkpoint and compressor ship as a single unit with the compressed cache as part of the model contract. The additional cost is one fine-tuning run, amortized over all subsequent serving. The underlying mechanism is functional robustness: the model routes around the discarded dimensions rather than reorganizing into the retained subspace. This principle should apply whenever the inference-time constraint is frozen and differentiable; quantization-aware and eviction-aware fine-tuning through frozen operators are natural extensions.

REPRODUCIBILITY STATEMENT

The supplementary material contains all code for calibration, CAFT training, the reimplemented baselines, and evaluation. The README provides the exact commands, seeds, and flags used to

produce each table and figure. Two included scripts recompute the seed statistics and regenerate Figures 2 and 3 from the raw outputs.

The method is fully specified in Section 3: the frozen basis and rank rule (Section 3.2, Equation 2), the projection (Equation 3), the straight-through estimator (Equation 4), and the matched-total-budget rule (Equation 5). Training recipes, seeds, evaluation protocols, and data splits are stated in Section 4.

All models are public checkpoints (Llama-3.2-1B, Llama-3.2-3B, Llama-3-8B, Llama-2-7B) and all datasets are public (WikiText-103, C4, OpenWebMath, peS2o, ARC, PIQA, HellaSwag, LongBench). RULER tasks are generated by the released code. Experiments ran on NVIDIA V100 32GB, RTX 2080 Ti 11GB, and A100 40GB GPUs under PyTorch 2.5.1 and Transformers 4.46.3. Training uses fp16 with fp32 master weights (full fine-tuning) or fp32 adapters (LoRA). Calibration and evaluation use bfloat16. The systems benchmark (Table 13) reports cache memory reduction and batch capacity on the A100.

Statement on the Use of AI

The authors used generative AI tools during this work. LLM-based coding assistants were used to assist with software implementation and experiment scaffolding, and generative AI tools were used to assist with manuscript drafting and editing. Generative AI was not used to formulate the research questions, design the proposed method or experimental protocol, analyze or interpret experimental results, or formulate the paper's scientific claims. All AI-assisted code and text were reviewed and verified by the authors. The authors take full responsibility for the final content of the paper.

REFERENCES

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2023.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench: A bilingual, multitask benchmark for long context understanding. In *Annual Meeting of the Association for Computational Linguistics*, 2024.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7432–7439, 2020.

Chi-Chih Chang, Wei-Cheng Lin, Chien-Yu Lin, Chong-Yan Chen, Yu-Fang Hu, Pei-Shuo Wang, Ning-Chi Huang, Luis Ceze, Mohamed S. Abdelfattah, and Kai-Chiang Wu. Palu: KV-cache compression with low-rank projection. In *International Conference on Learning Representations*, 2025.

Chi-Chih Chang, Wei-Cheng Lin, Chien-Yu Lin, Hung-Yueh Chiang, Yash Akhauri, Xilai Dai, Huiqiang Jiang, Yucheng Li, Luis Ceze, Kai-Chiang Wu, and Mohamed S. Abdelfattah. xKV: Cross-layer KV-cache compression via aligned singular vector extraction. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2026.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Yoav Gelberg, Yam Eitan, Michael Bronstein, Yarin Gal, and Haggai Maron. Training transformers for KV cache compressibility. *arXiv preprint arXiv:2605.05971*, 2026.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Geoffrey Hinton. Neural networks for machine learning. Coursera, Lecture 15b, 2012.

Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W. Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. KVQuant: Towards 10 million context length LLM inference with KV cache quantization. In *Advances in Neural Information Processing Systems*, 2024.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? In *Conference on Language Modeling*, 2024.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Tao Ji, Bin Guo, Yuanbin Wu, Qipeng Guo, Lixing Shen, Zhan Chen, Xipeng Qiu, Qi Zhang, and Tao Gui. Towards economical inference: Enabling DeepSeek's multi-head latent attention in any transformer-based LLMs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 33313–33328, 2025.

Hao Kang, Qingru Zhang, Souvik Kundu, Geonhwa Jeong, Zaoxing Liu, Tushar Krishna, and Tuo Zhao. GEAR: An efficient KV cache compression recipe for near-lossless generative inference of LLM. *arXiv preprint arXiv:2403.05527*, 2024.

Rahul Krishnan and Volker Schulz. Tensor decomposition of transformer key-value caches: Spectral structure and format comparison. *arXiv preprint arXiv:2609.28029*, 2026.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. SnapKV: LLM knows what you are looking for before generation. In *Advances in Neural Information Processing Systems*, 2024.

Bokai Lin, Zihao Zeng, Zipeng Xiao, Siqi Kou, Tianqi Hou, Xiaofeng Gao, Hao Zhang, and Zhijie Deng. MatryoshkaKV: Adaptive KV compression via trainable orthogonal projection. In *International Conference on Learning Representations*, 2025.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. KIVI: A tuning-free asymmetric 2bit quantization for KV cache. In *International Conference on Machine Learning*, 2024.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017.

Meta. Llama 3.2: Revolutionizing edge AI and vision with open, customizable models. https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/, 2024.

Keiran Paster, Marco Dos Santos, Zhangir Azerbayev, and Jimmy Ba. OpenWebMath: An open dataset of high-quality mathematical web text. *arXiv preprint arXiv:2310.06786*, 2023.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

Utkarsh Saxena, Gobinda Saha, Sakshi Choudhary, and Kaushik Roy. Eigen attention: Attention in low-rank space for KV cache compression. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 15332–15344, 2024.

Luca Soldaini and Kyle Lo. peS2o (pretraining efficiently on S2ORC) dataset. Technical report, Allen Institute for AI, 2023.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Luning Wang, Shiyao Li, Xuefei Ning, Zhihang Yuan, Shengen Yan, Guohao Dai, and Yu Wang. CSKV: Training-efficient channel shrinking for KV cache in long-context scenarios. In *NeurIPS ENLSP-IV Workshop*, 2024. arXiv:2409.10593.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, 2024.

Xianglong Yan, Zhiteng Li, Tianao Zhang, Haotong Qin, Linghe Kong, Yulun Zhang, and Xiaokang Yang. ReCalKV: Low-rank KV cache compression via head reordering and offline calibration. *arXiv preprint arXiv:2505.24357*, 2025.

Zhihang Yuan, Yuzhang Shang, Yue Song, Dawei Yang, Qiang Wu, Yan Yan, and Guangyu Sun. ASVD: Activation-aware singular value decomposition for compressing large language models. *arXiv preprint arXiv:2312.05821*, 2023.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, 2019.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. H_2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems*, 2023.

Yuhao Zhou, Sirui Song, Boyang Liu, Zhiheng Xi, Senjie Jin, Xiaoran Fan, Zhihao Zhang, Wei Li, and Xuanjing Huang. EliteKV: Scalable KV cache compression via RoPE frequency selection and joint low-rank projection. *arXiv preprint arXiv:2503.01586*, 2025.

APPENDIX

A C4 VALIDATION PERPLEXITY

Table 7: C4 validation perplexity at matched total cache budget (mirror of Table 1). PH-FT and CAFT cells are three-seed means; BASE and PH-BASE are deterministic.

Model	Budget	BASE	PH-BASE	FT	PH-FT	CAFT
Llama-3.2-1B	4×	13.23	526.78	20.00	1317.46	**124.52**
	2×		146.06		563.69	**46.82**
Llama-3.2-3B	4×	10.38	232.75	11.47	442.65	**96.64**
	2×		62.71		98.21	**27.82**
Llama-3-8B	4×	8.57	239.14	10.21	612.28	**110.21**
	2×		108.22		171.72	**44.80**

Table 8: C4 validation perplexity across six operators at 4× compression (mirror of Table 2). All cells are three-seed means.

	1B		3B		8B	
Operator	PH-FT	CAFT	PH-FT	CAFT	PH-FT	CAFT
ReCalKV-style	604.34	**59.74**	121.66	**37.97**	190.82	**54.73**
Palu-style	653.85	70.08	227.79	57.68	327.29	60.32
Energy-thr.	912.64	123.58	341.15	66.33	580.42	72.89
Whitened	690.05	112.95	441.19	80.90	673.15	91.47
Uniform	1317.46	124.52	442.65	96.64	612.28	110.21
xKV-style	1038.36	136.89	658.75	108.74	871.03	101.32

CAFT outperforms both PH-FT and PH-BASE in every C4 cell. ReCalKV-style achieves the lowest CAFT perplexity at every scale on C4, matching the WikiText ordering. xKV-style is the worst operator at 1B and 3B; at 8B Uniform is worst. The gap between CAFT and the uncompressed FT baseline is larger on C4 than on WikiText.

B CROSS-RATIO TRAINING AND EVALUATION MATRIX

Table 9 reports the 1B cross-ratio evaluation matrix. Rows index the training ratio; columns index the evaluation ratio at matched total cache budget.

The PH-FT baseline is a single row because the PH-FT arm has no training ratio: only the evaluation ratio selects its basis.

Table 9: Cross-ratio generalization, 1B WikiText-103 perplexity (Uniform, single run). Rows = training ratio, columns = evaluation ratio at matched budget.

Train \ Eval	2×	3×	4×	8×
2×	13.300	n/a	n/a	n/a
3×	19.165	16.597	n/a	n/a
4×	25.175	20.444	20.083	n/a
8×	45.128	42.355	41.480	41.034
PH-FT baseline	159.23	415.00	355.70	689.87

Two findings stand out. First, within each column the diagonal entry (matched training) is always the best, and quality degrades monotonically as the training ratio moves further from the evaluation ratio (6 of 6 orderings on both corpora). Second, even the worst mismatch measured (train 8×, eval 2×, a 4× budget mismatch) still beats the matched-budget PH-FT by 3.5× on WikiText. The penalty for a mismatched training ratio is graded and real, but the in-loop advantage is not a knife-edge that requires the exact deployment budget to be known at training time.

C TRUE-MHA ARCHITECTURE CONTROL (LLAMA-2-7B)

Every other model in this study uses grouped-query attention (GQA) with 8 KV heads. Llama-2-7B has true multi-head attention (MHA) with 32 KV heads, so the KV cache per token is $4\times$ larger at the same head dimension. Table 10 tests whether the CAFT result transfers off GQA.

Tokenizer non-comparability. Llama-2 uses a different tokenizer from Llama-3 (WikiText token count 335 635 vs. Llama-3's 288 937). Perplexity values are therefore not comparable across the two families; only the within-model CAFT vs. PH-FT ratios are meaningful, and the cross-architecture comparison to the GQA blocks is direction-only.

Evaluation max-length. These rows were evaluated at `max-length` = 1024 (the 7B training sequence length), while the Llama-3 blocks use `max-length` = 2048. This is an instrument difference, not a quality difference, and it does not affect within-model ratio comparisons.

Table 10: Llama-2-7B true-MHA perplexity (single run, seed 0). FT uncompressed baseline: WT 4.87 / C4 7.84. Ratio = PH-FT / CAFT.

Op.	Budget	CAFT WT	PH-FT WT	WT ratio	CAFT C4	PH-FT C4	C4 ratio
Uniform	4×	7.252	26.518	3.7×	24.106	141.69	5.9×
Uniform	3×	6.344	13.473	2.1×	18.229	64.55	3.5×
Uniform	2×	5.518	7.206	1.3×	12.735	19.95	1.6×
ReCalKV-style	4×	5.246	6.823	1.3×	11.158	14.38	1.3×
ReCalKV-style	3×	5.097	6.210	1.2×	9.872	12.07	1.2×
ReCalKV-style	2×	4.956	5.514	1.1×	8.570	9.58	1.1×

The qualitative pattern transfers: CAFT beats PH-FT in all 12 cells, the gap widens with the ratio, and ReCalKV-style (Yan et al., 2025) post-hoc stays relatively controlled while Uniform post-hoc collapses at $4\times$, the same ordering observed under GQA at 3B and 8B.

D COMPRESSION-AWARE COMPETITORS AT 8B

Table 11: Llama-3-8B WikiText-103 and C4 perplexity at matched total cache bytes. CAFT and PH-FT use the Uniform SVD operator. Competitor rows are single run (seed 0); PH-FT and CAFT are three-seed means. Uncompressed FT: WikiText 5.49 / C4 10.21.

	WikiText-103			C4		
Arm	2×	3×	4×	2×	3×	4×
Learned projection (CSKV-style)	6.68	8.68	11.09	29.73	58.87	72.29
MatryoshkaKV	8.07	9.68	11.10	19.48	27.92	36.19
MHA2MLA	7.60	9.28	10.52	28.75	62.64	75.32
GEAR (post-hoc quantization)	**5.50**	**5.57**	**5.61**	**10.15**	**10.31**	**10.44**
PH-FT (Uniform SVD)	11.17	30.95	54.19	171.72	456.52	612.28
CAFT (Uniform SVD)	6.79	8.36	10.11	44.80	72.35	110.21

Table 11 compares CAFT at 8B against four published compression methods at matched total cache bytes.[2] The CAFT and PH-FT rows use the Uniform SVD operator, the simplest and weakest of the six operators in Table 2. CAFT's best operator, ReCalKV-style, reaches 7.01 WikiText at $4\times$, below every low-rank competitor in this table (10.52 to 11.10); the $2\times$ and $3\times$ ReCalKV cells were not measured at 8B. In principle, any of the competitor operators could also be trained through the CAFT loop.

Downstream accuracy is not compared head-to-head because the competitors use different checkpoint lineages and training corpora, confounding compression quality with lineage effects. Competitor cells are single-seed (seed 0); the PH-FT and CAFT rows are three-seed means.

[2] GEAR's $3\times$ cell stores 42936 bytes per token against 43008 for the low-rank arms, making it slightly more compressed at that budget.

E STACKING WITH 4-BIT CODE QUANTIZATION

The retained low-rank codes are quantized to 4 bits in both arms, raising the effective cache ratio to $8\times$ through $16\times$ depending on the base compression ratio. PH-FT collapses to near-chance downstream accuracy and triple-digit perplexity at every budget, while CAFT remains functional throughout.

Table 12: CAFT stacked with 4-bit code quantization (Llama-3.2-1B, Uniform operator, three-seed means). The difference is significant in every cell (paired t, $n = 3$, all $p < 0.005$). The 4-bit quantization is simulated.

	WikiText-103 PPL		C4 PPL		Downstream Acc.	
Budget	PH-FT	CAFT	PH-FT	CAFT	PH-FT	CAFT
$2\times$ / $8\times$	402.86	**12.89**	1010.75	**52.62**	0.326	**0.453**
$3\times$ / $12.2\times$	1414.10	**17.06**	3029.83	**106.81**	0.317	**0.407**
$4\times$ / $16\times$	665.17	**21.55**	1278.19	**163.46**	0.316	**0.380**

F SYSTEMS COST DETAILS

Table 13: Systems cost at $4\times$ compression. *Top:* KV-cache memory reduction, confirmed by a real compressed cache. *Middle:* prefill overhead on A100 (both arms run the same attention kernel). *Bottom left:* batch capacity at context 4096. *Bottom right:* offline costs (1B).

KV-cache memory (real compressed cache, 1B $4\times$)

Context	Uncompressed	Compressed	Reduction
512	18.87 MB	4.72 MB	4.00×
2048	69.21 MB	17.32 MB	4.00×
4096	136.31 MB	34.11 MB	4.00×

Prefill overhead, $4\times$ Uniform, A100, bf16, batch 1

	@512	@2048	@4096
1B	−15.4%	+1.1%	+1.0%
8B	−2.7%	+1.2%	+1.3%

Batch capacity (A100, ctx 4096)	*Offline costs (1B)*
1B: 32 → 64	Calibration: ∼2.6 / 4.2 min
8B: 8 → 16	Training: ∼150× cal.