NORMSUB: NORMALIZATION-AFFINE SUBSPACE ADAPTATION FOR QUANTIZATION RECOVERY

Anonymous authors
Paper under double-blind review

ABSTRACT

Low-bit quantization can substantially degrade model accuracy. Beyond improving quantizers and weight reconstruction, recovery requires understanding which parameters are useful to adapt under a limited budget. We investigate normalization-affine parameters as a structured, low-dimensional update subspace. In a fixed-graph study with matched parameter counts, data, and update budgets, affine updates achieve lower perplexity and higher mean accuracy than the tested weight-coordinate controls, with a small advantage over gradient-selected coordinates. Functional diagnostics further show that task-level recovery need not reduce every local reconstruction error. These findings motivate NormSub, a two-stage affine adaptation framework around a target post-training quantization (PTQ) backend. The pre-stage adapts only normalization-affine parameters on a disposable backend-aligned quantized proxy, then discards the proxy, restores the original floating-point weights, and transfers only the backend-compatible affine state for fresh PTQ reconstruction. The post-stage refines the affine coordinates exposed by the reconstructed graph while freezing its non-affine persistent state. Across the evaluated dense language models and PTQ backends, NormSub improves perplexity and mean zero-shot accuracy. On Qwen3-8B W2A4KV4 with QuaRot+ResComp, the unweighted six-task mean accuracy increases from 49.6% to 58.1%.

1 INTRODUCTION

Quantization is a practical approach to model compression for resource-constrained deployment, reducing the precision of model parameters and activations (Jacob et al., 2018). Post-training quantization (PTQ) is particularly appealing because it converts pretrained models to low precision without full retraining (Li et al., 2021a; Frantar et al., 2023; Li et al., 2025; 2026). Existing PTQ methods reduce quantization error through optimized rounding and weight reconstruction (Nagel et al., 2020; Frantar et al., 2023; Li et al., 2025; 2026), as well as transformations of weight and activation distributions (Xiao et al., 2023; Ashkboos et al., 2024). At extremely low bit widths, however, rounding and clipping distortions can still substantially degrade task performance (Wei et al., 2022). Beyond improving quantization itself, we ask how the choice of trainable parameter coordinates affects performance recovery under a limited adaptation budget.

Normalization-affine parameters are a natural candidate for such adaptation. Their scales and, where present, biases are shared across tokens or spatial locations, allowing a small parameter set to influence downstream feature responses (Ioffe & Szegedy, 2015; Zhang & Sennrich, 2019). Prior work supports their use in quantization: Norm Tweaking improves low-bit language models by updating normalization parameters to align channel-wise activation distributions with a floating-point reference (Li et al., 2024). In vision, PTQ4VM uses global affine calibration to compensate for accumulated statistical distortions associated with Batch Normalization (Zhu et al., 2026). To examine how coordinate choice affects recovery, we compare normalization-affine coordinates with selected weight-coordinate subsets. This fixed-graph comparison matches trainable parameter counts, data, objectives, and update budgets. Affine updates achieve lower perplexity and higher mean accuracy than the tested weight-coordinate controls, with a small advantage over gradient-selected weights (Figure 3). Functional diagnostics further show that task-level recovery need not coincide with uniformly lower local reconstruction error (Figure 4). Together, these observations support normalization-affine parameters as an effective, structured, low-dimensional correction subspace.

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.

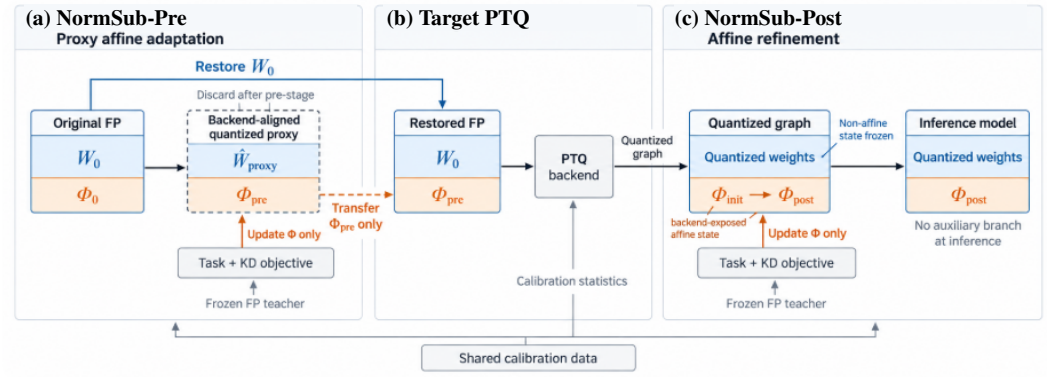


Figure 1: **Overview of NormSub.** Pre learns only normalization-affine coordinates on a backend-aligned quantized proxy, which is then discarded. Reconstruction restores $\mathbf{W}_0$ and transfers only $\mathbf{\Phi}_{\text{pre}}$ under the backend's coordinate convention. Post freezes non-affine persistent state and refines $\mathbf{\Phi}_{\text{init}} = \phi(\mathcal{G}_2)$ to $\mathbf{\Phi}_{\text{post}}$; $\mathbf{\Phi}_{\text{init}}$ need not equal $\mathbf{\Phi}_{\text{pre}}$. No auxiliary branch remains at inference. ImageNet's observer-recalibration exception is specified in Appendix C.

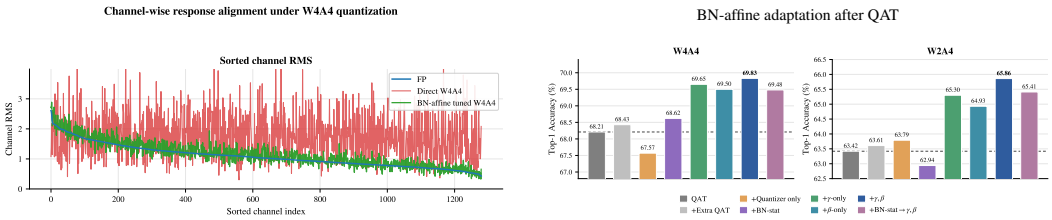
This subspace view motivates NormSub, a two-stage affine adaptation framework around a compatible PTQ backend (Figure 1). Its pre-stage learns an affine state on a disposable backend-aligned quantized proxy. The original floating-point weights are then restored, and only the learned, backend-compatible affine state is transferred for PTQ reconstruction. The post-stage refines the affine coordinates exposed by the reconstructed quantized graph. The two stages respectively adapt the state presented to PTQ and correct the resulting quantized model. Both stages optimize only normalization-affine coordinates, and no learned auxiliary branch is added at inference.

Our contributions are summarized as follows:

- **A controlled study of quantization recovery subspaces.** We compare normalization-affine coordinates with selected equal-size weight-coordinate subsets under matched adaptation budgets, showing that coordinate structure matters for recovery. Functional diagnostics and local response analysis further characterize the effects of these updates on downstream model responses.
- **A two-stage affine adaptation framework.** We propose NormSub, which couples affine adaptation before PTQ reconstruction with refinement of the backend-exposed affine state afterward.
- **Validation across backends, tasks, and model families.** We evaluate the complete pipeline on dense language models and vision tasks across multiple PTQ backends and low-bit settings, and examine post-only adaptation on native mixture-of-experts (MoE) models and quantization-aware training (QAT) checkpoints. On Qwen3-8B with QuaRot+ResComp at W2A4KV4, mean zero-shot accuracy over six tasks increases from 49.6% to 58.1%.

2 Related Work

Quantization reconstruction and error compensation. Reconstruction-based PTQ uses optimized rounding to approximate floating-point layer or block responses (Nagel et al., 2020; Li et al., 2021a), while GPTQ uses approximate second-order information for weight quantization (Frantar et al., 2023). GPTAQ addresses input discrepancies caused by preceding quantized layers through asymmetric calibration (Li et al., 2025). ResComp further accounts for the difference between compensated and original weights when forming compensation residuals (Li et al., 2026). NormSub complements these approaches with normalization-affine adaptation around PTQ reconstruction while retaining the selected backend's quantization and weight-compensation rules.



(a) BN-affine adaptation substantially reduces channel-wise RMS mismatch under W4A4.

(b) **Residual directions after QAT.** Affine-only refinement remains useful from the same LSQ+ checkpoint.

Figure 2: **Affine correction in PTQ and post-QAT controls.** The shared-checkpoint comparison tests residual correction under a fixed budget, not asymptotic superiority over full-space training.

Distribution transformations for quantization. SmoothQuant uses equivalent channel scaling to redistribute quantization difficulty from activations to weights (Xiao et al., 2023). QuaRot applies Hadamard rotations to reduce outliers, while SpinQuant learns rotations for the quantized model (Ashkboos et al., 2024; Liu et al., 2025). FlatQuant learns affine transformations for individual linear layers to flatten weight and activation distributions (Sun et al., 2025). These transformation parameters differ from the normalization-affine coordinates studied here. During the LLM affine-adaptation stages of NORMSUB, we do not optimize the selected rotations or quantizer parameters; updates are restricted to backend-exposed normalization-affine coordinates.

Normalization-based quantization recovery. Norm Tweaking demonstrates the value of restricting adaptation to normalization parameters (Li et al., 2024). In its GPTQ-based formulation, it interleaves blockwise quantization with normalization updates that match the channelwise means and variances of quantized and floating-point block outputs. NORMSUB likewise restricts its LLM adaptation to normalization-affine coordinates, but uses task and prediction-level knowledge-distillation objectives on completed quantized graphs. Following blockwise initial quantization, PTQ4VM jointly optimizes layerwise scale–shift corrections to quantized, BatchNorm-folded convolutional weights and activation quantization scales using a task-level matting objective (Zhu et al., 2026). The complete NORMSUB pipeline additionally transfers only the learned, backend-compatible affine state from a disposable proxy to the restored floating-point model for fresh PTQ reconstruction. It then refines the affine coordinates exposed by the reconstructed graph.

Structured low-dimensional adaptation. Bias-only tuning and low-rank adaptation reduce trainable parameters through coordinate restriction and structured weight updates, respectively (Ben Zaken et al., 2022; Hu et al., 2022). Training only BatchNorm affine parameters has also been compared with training equally many randomly selected coordinates, with the remaining weights fixed at random initialization (Frankle et al., 2021). TENT estimates normalization statistics and optimizes affine parameters through test-time entropy minimization (Wang et al., 2021). Our controlled study examines recovery of a pretrained model on a fixed quantized graph under a limited adaptation budget. By matching trainable parameter counts, data, objectives, and update budgets, we compare the recovery achieved by normalization-affine and selected weight-coordinate subspaces.

3 METHOD

We study quantization recovery through normalization-affine coordinates and use this structured update subspace before and after PTQ reconstruction. We first define the adaptation problem, examine the coordinate controls and functional diagnostics, and then describe the two-stage NORMSUB construction.

3.1 QUANTIZATION RECOVERY IN A STRUCTURED UPDATE SUBSPACE

Update coordinates. Let $\mathcal{S}_0 = (\mathbf{W}_0, \mathbf{\Phi}_0)$ denote the original floating-point checkpoint. For a given quantized computation graph $\mathcal{G}$, we group its model parameters and persistent quantizer set-

tings as $\Theta = (\mathbf{W}, \mathbf{\Phi}, \mathbf{\Omega})$. Here, $\mathbf{W}$ contains weights and biases outside the normalization-affine coordinates, $\mathbf{\Phi}$ contains the normalization-affine coordinates exposed by $\mathcal{G}$, and $\mathbf{\Omega}$ contains persistent quantizer parameters and settings. This grouping identifies update coordinates rather than enumerating all execution state. For BatchNorm (Ioffe & Szegedy, 2015), $\mathbf{\Phi}$ includes channelwise scales and biases; for RMSNorm (Zhang & Sennrich, 2019), it includes scales only.

Normalization-affine adaptation restricts parameter updates to

$$\mathcal{S}_{\mathbf{\Phi}} = \{\Delta\Theta : \Delta\mathbf{W} = \mathbf{0},\ \Delta\mathbf{\Omega} = \mathbf{0}\}. \tag{1}$$

This restriction defines the allowed updates in the affine coordinate system exposed by $\mathcal{G}$. The parameterization of these coordinates may depend on the backend construction.

For the LLM affine stages and affine-only fixed-graph controls, all non-affine persistent state is fixed, including backbone weights and biases, router parameters, rotations, quantizer parameters and settings, and buffers such as BatchNorm running statistics. Input-dependent activations, quantization scales computed at runtime, and expert assignments may still vary. The ImageNet extension additionally recalibrates observers between tuning epochs, separately from gradient-based affine adaptation, and therefore does not keep quantizer state fixed across epochs (Appendix C).

Recovery objective. For an input $\boldsymbol{x}$ with task target y, let $\boldsymbol{z}_q$ and $\boldsymbol{z}_0$ denote the output logits of the quantized student and the frozen original floating-point teacher, respectively:

$$\begin{aligned} \boldsymbol{z}_q &= \widetilde{f}_{\mathcal{G}}(\boldsymbol{x}; \mathbf{\Phi}), \\ \boldsymbol{z}_0 &= f(\boldsymbol{x}; \mathbf{W}_0, \mathbf{\Phi}_0). \end{aligned} \tag{2}$$

We combine task supervision with prediction-level knowledge distillation (Hinton et al., 2015). Let $p_\tau(\boldsymbol{z}) = \mathrm{softmax}(\boldsymbol{z}/\tau)$, where τ is the distillation temperature. The core recovery objective is

$$\begin{aligned} \mathcal{L}_{\mathrm{NormSub}} = \mathcal{L}_{\mathrm{task}}(\boldsymbol{z}_q, y) \\ + \lambda_{\mathrm{KD}} \tau^2 \,\mathrm{KL}(p_\tau(\boldsymbol{z}_0) \,\|\, p_\tau(\boldsymbol{z}_q)). \end{aligned} \tag{3}$$

Here, λ_{KD} weights the distillation term. The task loss is next-token negative log-likelihood for language modeling, classification cross-entropy for image classification, and pixelwise cross-entropy for semantic segmentation. Some vision protocols additionally use the channel-scale regularizer specified in Appendix C.

Finite-step affine adaptation. Let $\phi(\mathcal{G})$ extract the normalization-affine state exposed by $\mathcal{G}$. For the fixed-state setting above, $\mathcal{A}_T(\mathcal{G}; \mathbf{\Psi})$ returns the affine state obtained from a graph-compatible initialization $\mathbf{\Psi}$ after T updates under the prescribed recovery objective. The updates use Eq. (3), with protocol-specific regularization only where stated above, and follow the prescribed data and optimization protocol. It denotes finite-step adaptation, not an exact minimizer.

3.2 Evidence for Normalization-Affine Recovery

Matched-coordinate comparison. We test whether normalization-affine coordinates provide effective recovery relative to equally sized weight-coordinate subsets. The comparison starts from the same completed Qwen3-8B W2A4KV4 QuaRot–ResComp quantized backbone. Each trainable branch updates exactly 299,008 scalar coordinates: normalization-affine parameters, uniformly sampled weight coordinates (RandomW), gradient-selected coordinates (TopGradW), or coordinates restricted to attention (AttentionW) and feed-forward projections (MLPW). TopGradW ranks calibration-gradient magnitudes within a uniformly sampled candidate pool. The selected weight masks remain fixed throughout adaptation. The Frozen branch performs no updates.

The weight-coordinate controls learn floating-point sparse increments over frozen quantized base operators. Both the base operator and its correction receive the same input after any prescribed rotation and activation quantization. The increments are stored in FP32 and cast to the input dtype during forward execution. Their learned contributions are retained at evaluation without projection onto the original weight-quantization grid. This comparison therefore tests coordinate adaptation on a shared quantized backbone, rather than updates constrained to a common low-bit weight grid.

All trainable branches share the calibration blocks, task-plus-KD objective, 200-update budget, batch size, sequence length, optimizer, and prescribed learning-rate settings (Appendix A). Figure 3 shows that affine updates achieve lower perplexity and higher mean accuracy than the tested

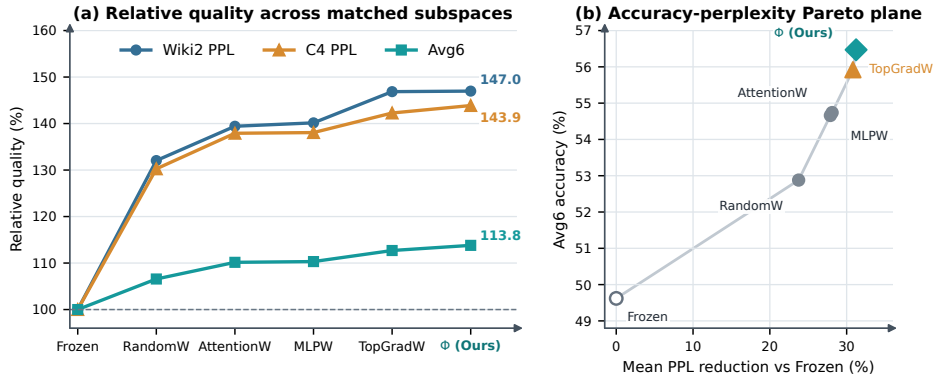
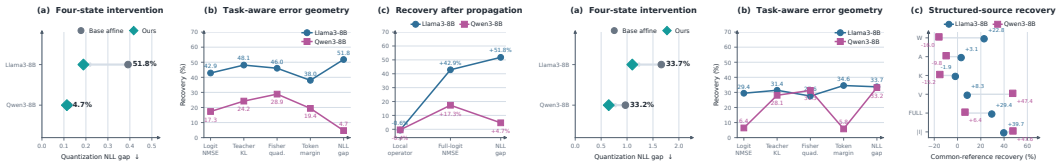


Figure 3: **Matched-budget subspaces on Qwen3-8B W2A4KV4.** Trainable branches share $|\mathbf{\Phi}|$, calibration data, update count, and objective. (a) Relative quality, normalized to Frozen = 100; higher is better. (b) Avg6 versus mean relative WikiText-2/C4 perplexity reduction. $\mathbf{\Phi}$ achieves the strongest overall recovery among these controls.



(a) **W3A16 with GPTQ.** (b) **W2A4KV4 with QuaRot+ResComp.**

Figure 4: **Functional effects of post-only affine correction.** Four-state interventions and task-aware metrics compare Llama3-8B and Qwen3-8B. The final panels examine propagation under W3A16 and structured error sources under W2A4KV4. *Ours* denotes NORMSUB-POST, not the complete pipeline; functional recovery need not lower every local operator error.

weight-coordinate controls, with a small advantage over TopGradW. Because trainable dimensionality and update budgets are matched, the observed differences show that parameter count alone does not determine recovery in this setting. The results support normalization-affine coordinates as an effective update choice among the tested controls under the prescribed finite budget.

Complementary component diagnostics. The vision diagnostics examine related effects on existing graphs. Figure 2a shows that BN-affine tuning reduces the mismatch in the plotted channel-RMS profiles relative to the floating-point reference. Figure 2b shows that affine refinement can improve accuracy from an existing LSQ+ checkpoint under the reported additional-update budget. These observations complement the matched-coordinate study by illustrating channel-response adjustment and post-QAT recovery. They evaluate affine adaptation as a component, rather than the complete two-stage construction introduced below.

3.3 A LOCAL INTERPRETATION OF AFFINE RESPONSE CONTROL

Broadcast response control. Holding normalized features fixed, the affine response of channel c at layer ℓ satisfies

$$\begin{aligned} \boldsymbol{a}_{\ell,c} &= \gamma_{\ell,c}\boldsymbol{h}_{\ell,c} + \beta_{\ell,c}\mathbf{1}, \\ \Delta\boldsymbol{a}_{\ell,c} &= \boldsymbol{h}_{\ell,c}\Delta\gamma_{\ell,c} + \mathbf{1}\Delta\beta_{\ell,c}. \end{aligned} \tag{4}$$

Here, $\boldsymbol{h}_{\ell,c}$ collects the normalized responses across tokens or spatial locations. The bias terms are absent for RMSNorm. Thus, a small number of shared affine coordinates can affect many responses within a channel and their downstream computation. Under the fixed-statistics BN assumptions in Appendix B, these coordinates can exactly cancel a structured local scale–shift perturbation, while leaving an unstructured residual. RMSNorm instead provides multiplicative response control and does not generally cancel additive offsets.

Local output-response geometry. For a graph $\mathcal{G}$ with fixed non-affine persistent state, let $\mathbf{\Phi}_{\mathcal{G},0} = \phi(\mathcal{G})$ denote its initial affine coordinates. Consider the output logits on fixed inputs, with input dependence suppressed for clarity. Under a local differentiable approximation of the graph, a small affine update induces

$$\begin{aligned} \boldsymbol{z}_q(\mathbf{\Phi}_{\mathcal{G},0} + \Delta\mathbf{\Phi}) \approx \boldsymbol{z}_q(\mathbf{\Phi}_{\mathcal{G},0}) \\ + \mathbf{J}_{\mathbf{\Phi}}\Delta\mathbf{\Phi}, \end{aligned} \tag{5}$$

where $\mathbf{J}_{\mathbf{\Phi}}$ is the logit-response Jacobian of this local approximation at $\mathbf{\Phi}_{\mathcal{G},0}$. Define the teacher–student residual as

$$\boldsymbol{e}_z = \boldsymbol{z}_0 - \boldsymbol{z}_q(\mathbf{\Phi}_{\mathcal{G},0}).$$

For a fixed positive-semidefinite output-space weighting $\mathbf{H}_z$, we consider the response-matching surrogate

$$\min_{\Delta\mathbf{\Phi}} \|\boldsymbol{e}_z - \mathbf{J}_{\mathbf{\Phi}}\Delta\mathbf{\Phi}\|^2_{\mathbf{H}_z}, \qquad \mathbf{H}_z \succeq 0, \tag{6}$$

where $\|\boldsymbol{v}\|^2_{\mathbf{H}_z} = \boldsymbol{v}^\top \mathbf{H}_z \boldsymbol{v}$.

The parameter subspace $\mathcal{S}_{\mathbf{\Phi}}$ specifies which coordinates may change, whereas $\mathrm{range}(\mathbf{J}_{\mathbf{\Phi}})$ describes their first-order output-response directions. The surrogate measures how well the residual can be represented by these directions under the weighting $\mathbf{H}_z$. Directions in the null space of $\mathbf{H}_z$ are unpenalized. This provides a local interpretation of response capacity, rather than an exact expansion of Eq. (3) or the objective optimized by the finite-step operator $\mathcal{A}_T$. The corresponding weighted-projection analysis and its limits are given in Appendix B.

Local error and functional recovery. Figure 4 examines post-only affine correction on Llama3-8B and Qwen3-8B. The probes compare graph-local initial and adapted affine states using fixed inputs and a common original floating-point reference. They track local operator discrepancies, full-model logit errors, and the NLL gap. The W2A4KV4 analysis additionally evaluates weight, activation, key-cache, and value-cache perturbations separately and jointly (Appendix A).

The diagnostics show that downstream logit and NLL recovery can occur even when some local operator errors increase. The W2A4KV4 diagnostics also report non-additive effects across the individual and combined perturbation sources. These observations are consistent with recovery through downstream propagation and interaction, without identifying a unique causal mechanism. They motivate evaluating task-level recovery alongside local reconstruction metrics rather than requiring every local error to decrease.

3.4 NormSub: Adaptation around PTQ Reconstruction

We use normalization-affine adaptation at two points in the quantization workflow: before PTQ reconstruction to adjust the state presented to the backend, and afterward to refine the resulting quantized graph (Figure 1).

Backend-dependent construction. Let $\mathcal{B}_{\mathcal{D}}(\mathcal{S}_0)$ denote the target backend's complete PTQ construction from the original floating-point checkpoint using calibration data $\mathcal{D}$. We write $\mathcal{B}_{\mathcal{D}}(\mathcal{S}_0; \mathbf{\Psi})$ for a fresh construction that transfers a compatible affine state $\mathbf{\Psi}$ into the backend-prepared floating-point graph before completing PTQ. The transfer follows the backend's affine-coordinate convention; it does not require the resulting quantized graph to expose an unchanged copy of $\mathbf{\Psi}$.

For unfused GPTQ, the transferred coordinates are the original normalization scales. In the evaluated rotated W2 implementation, the original scales are folded into projections, and the exposed residual-normalization affine coordinates initially have identity values. The learned state is transferred into these coordinates of a freshly fused-and-rotated graph, rather than interpreted as the original checkpoint's scales. Restoring $\mathbf{W}_0$ therefore specifies the source checkpoint for reconstruction, not numerical equality of internal weights after backend fusion or rotation.

Two-stage state flow. For the fixed-state dense-LLM setting, both PTQ constructions use the same calibration tensor $\mathcal{D}$. Affine adaptation uses the prescribed subblocks from this tensor under the protocol in Appendix A. Using the adaptation operator from Section 3.1, the complete construction is

Algorithm 1 NORMSUB: normalization-affine tuning around PTQ reconstruction

Require: Immutable FP checkpoint $\mathcal{S}_0$; PTQ backend $\mathcal{B}$; calibration tensor $\mathcal{D}$; update budgets T_{pre} and T_{post}
Ensure: Final quantized graph $\mathcal{G}_{\text{NormSub}}$
1: **NormSub-Pre: normalization-affine tuning**
2: Construct the backend-aligned quantized proxy $\mathcal{G}_1$ and learn $\mathbf{\Phi}_{\text{pre}}$ with T_{pre} affine-only updates (Eq. (7a)).
3: Retain only $\mathbf{\Phi}_{\text{pre}}$ and discard the proxy.
4: **Target PTQ: restore, transfer, and reconstruct**
5: Restore $\mathcal{S}_0$ and reconstruct $\mathcal{G}_2$ with affine-only transfer of $\mathbf{\Phi}_{\text{pre}}$ (Eq. (7b)).
6: Set $\mathbf{\Phi}_{\text{init}} \leftarrow \phi(\mathcal{G}_2)$.
7: **NormSub-Post: normalization-affine tuning**
8: Starting from $\mathbf{\Phi}_{\text{init}}$, learn $\mathbf{\Phi}_{\text{post}}$ on $\mathcal{G}_2$ with T_{post} affine-only updates (Eq. (7c)).
9: **return** $\mathcal{G}_{\text{NormSub}}$ defined by Eq. (7d), with all parameters frozen.

$$\begin{aligned} \mathcal{G}_1 &= \mathcal{B}_{\mathcal{D}}(\mathcal{S}_0), \\ \mathbf{\Phi}_{\text{pre}} &= \mathcal{A}_{T_{\text{pre}}}\big(\mathcal{G}_1; \phi(\mathcal{G}_1)\big). \end{aligned} \tag{7a}$$

$$\begin{aligned} \mathcal{G}_2 &= \mathcal{B}_{\mathcal{D}}(\mathcal{S}_0; \mathbf{\Phi}_{\text{pre}}), \\ \mathbf{\Phi}_{\text{init}} &= \phi(\mathcal{G}_2). \end{aligned} \tag{7b}$$

$$\mathbf{\Phi}_{\text{post}} = \mathcal{A}_{T_{\text{post}}}(\mathcal{G}_2; \mathbf{\Phi}_{\text{init}}). \tag{7c}$$

$$\mathcal{G}_{\text{NormSub}} = \mathcal{G}_2[\mathbf{\Phi} \leftarrow \mathbf{\Phi}_{\text{post}}]. \tag{7d}$$

NORMSUB-PRE optimizes only normalization-affine coordinates on the disposable backend-aligned quantized proxy $\mathcal{G}_1$. In the evaluated dense order experiments, the target backend itself constructs this proxy, which is executed in quantize–dequantize form. After adaptation, only $\mathbf{\Phi}_{\text{pre}}$ is retained from the proxy; its quantized weights, quantizer state, and optimizer state are discarded.

Pre adjusts the state supplied to PTQ through affine-state transfer, whereas Post refines the affine coordinates exposed by the reconstructed graph. Algorithm 1 summarizes the fixed-state dense-LLM procedure. Both stages optimize only normalization-affine coordinates using Eq. (3), with all non-affine persistent state held fixed. Both stages use the original floating-point model as the frozen teacher, and each initializes a fresh optimizer.

Scope, cost, and inference. The ImageNet extension follows the same stage ordering but includes the separate observer-recalibration procedure described in Appendix C; these observer updates are not part of the fixed-state operator $\mathcal{A}_T$ above. The MoE and post-QAT experiments instead apply Post alone to existing checkpoints and do not evaluate the complete two-stage pipeline.

Without reusing a compatible completed PTQ baseline as $\mathcal{G}_1$, the full construction requires two PTQ passes and two affine adaptation stages. The order ablation in Table 3 matches per-stage update budgets, not total computation across variants. Only affine parameters require optimizer state, but the frozen model and backward activations still contribute to adaptation cost.

The final model retains the reconstructed backend graph with $\mathbf{\Phi}_{\text{post}}$. The proxy, teacher, and adaptation objective are used only offline; no learned auxiliary branch is retained at inference.

4 EXPERIMENTS

Settings. Dense experiments use Llama2-7B (Touvron et al., 2023), Llama3-8B (Llama team, 2024), and Qwen3-8B (Yang et al., 2025). Calibration uses 128 training sequences of 2,048 tokens (seed 0): C4 (Raffel et al., 2020) for W3A16 and WikiText-2 (Merity et al., 2017) for W2A4KV4.

Weights use symmetric groups of 128; W2 additionally uses clipping and dynamic per-token asymmetric A4/K4/V4. Paired runs share checkpoints, tokenizers, calibration indices, quantization settings, and evaluation code. Each dense affine stage uses 200 AdamW updates, batch size 1, sequence length 256, and a task-plus-KD objective. We report WikiText-2/C4 perplexity and the unweighted six-task zero-shot accuracy. NORMSUB denotes the complete pipeline; NORMSUB-POST denotes post-only adaptation. Appendices A and C give the detailed dense, MoE, and vision settings.

4.1 CROSS-DOMAIN AND CHECKPOINT CONTROLS

Classification, segmentation, and post-QAT adaptation. Tables 1 and 2 evaluate the complete RTN- and QDrop-backed pipelines. Both improve their baselines at both precisions; the QDrop-backed pipeline gives the best W3A4 result in each table. ImageNet uses full-training-set adaptation and epochwise observer recalibration. The separate post-QAT controls improve several checkpoints, but OOQ-Freeze W2A4 drops from 67.42% to 62.81% (Table 8, Appendix C).

Model	Method	FP32	Quant. Acc. ↑	
			W4A4	W3A4
MobileNetV2	RTN	71.87	0.33	0.11
	NORMSUB (RTN)		**66.11**	51.07
	QDrop		60.71	51.75
	NORMSUB (QDrop)		62.91	**53.45**

Table 1: **ImageNet-1K accuracy (%), MobileNetV2.** Highlighted rows use full NORMSUB, with full-training-set adaptation and epochwise observer recalibration. Bold marks the best quantized result per precision.

Model	Method	FP32	Quantized mIoU ↑	
			W4A4	W3A4
U-Net	RTN	68.52	3.27	1.24
	NORMSUB (RTN)		**67.21**	59.07
	QDrop		66.45	58.21
	NORMSUB (QDrop)		66.78	**61.36**

Table 2: **Cityscapes mIoU (%), U-Net.** Highlighted rows use full NORMSUB with the stated backend. Bold marks the best quantized result per precision.

Stage ablation. Table 3 separates Pre, Post, and the complete pipeline. The complete pipeline gives the lowest perplexities in all four settings and the highest Avg6 in three; Llama3-8B W3A16 slightly favors Post alone. The 200-update budget is matched per affine stage, not in total computation.

Table 3: **Stage ablation.** Each affine stage uses 200 updates; full NORMSUB uses both stages and a second PTQ (GPTQ) pass, not a matched total budget. Wiki2/C4 are perplexities; Avg6 is mean accuracy (%). Bold marks the best value within each setting.

Order	Llama3-8B						Qwen3-8B					
	W3A16			W2A4KV4			W3A16			W2A4KV4		
	Wiki2↓	C4↓	Avg6↑	Wiki2↓	C4↓	Avg6↑	Wiki2↓	C4↓	Avg6↑	Wiki2↓	C4↓	Avg6↑
PTQ	8.33	13.10	68.83	18.07	50.56	46.81	11.29	17.09	71.87	18.71	38.27	49.62
PTQ + NORMSUB-POST	7.89	12.21	**70.08**	12.83	30.35	52.49	10.60	16.38	72.17	12.73	26.80	56.47
NORMSUB-PRE + PTQ reconstruction	8.28	12.87	68.89	13.93	37.96	48.72	10.74	16.42	72.31	12.99	27.99	55.28
NORMSUB	**7.87**	**12.13**	70.05	**12.31**	**28.86**	**53.44**	**10.51**	**16.15**	**72.34**	**12.03**	**25.37**	**58.05**

4.2 LOW-BIT LANGUAGE-MODEL RECOVERY

Complete dense-LLM pipeline. Tables 4 and 6 report the complete results at W3A16 and W2A4KV4. Across the reported model–backend pairs, the complete NORMSUB pipeline improves both perplexities and mean zero-shot accuracy, although individual tasks can decline. On Qwen3-8B with QuaRot+ResComp at W2A4KV4, mean accuracy increases from 49.6% to 58.1%. The learned-rotation extension and dense post-only controls are reported in Appendices E and F, respectively.

Post-only MoE extension. Tables 5 and 7 evaluate NORMSUB-POST on existing quantized backbones, with router parameters and KV caches unquantized. Both perplexities and mean accuracy improve in the reported pairs, while some individual task scores decline. These comparisons test post-only refinement rather than the complete two-stage pipeline or a specific routing mechanism.

Table 4: **W3A16 results on dense LLMs (group size 128).** Highlighted rows use the complete NORMSUB pipeline. Avg. is the six-task mean; ResComp* uses GPTAQ as its base quantizer. Bold marks the best quantized result.

Model	Method	Perplexity ↓		Zero-shot Accuracy (%) ↑						Avg. ↑
		Wiki2	C4	PIQA	ARC-E	ARC-C	HS	WG	BoolQ	
Llama2-7B	FP16	5.47	7.27	78.9	74.6	46.1	75.9	69.2	77.7	70.4
	RTN	8.22	11.53	76.3	66.5	40.6	68.7	65.8	67.5	64.2
	NORMSUB (RTN)	7.40	10.29	77.4	68.9	41.9	71.0	66.2	70.9	66.1
	AWQ	6.79	8.93	77.1	69.9	41.8	71.2	67.7	71.5	66.3
	NORMSUB (AWQ)	6.47	8.55	77.5	**70.8**	**43.1**	72.4	67.3	72.9	**67.3**
	GPTQ	6.75	13.72	76.8	66.5	39.7	68.3	67.5	68.9	64.6
	NORMSUB (GPTQ)	6.28	8.31	77.3	66.3	41.6	72.1	**67.8**	69.4	65.8
	GPTAQ	6.81	8.40	77.8	69.2	40.7	71.9	67.6	71.5	66.5
	NORMSUB (GPTAQ)	6.27	8.16	**78.3**	70.5	40.8	72.3	66.7	71.3	66.7
	ResComp*	6.25	8.19	77.4	69.5	41.5	72.3	67.4	71.5	66.6
	NORMSUB (ResComp*)	**6.18**	**8.12**	77.5	69.6	42.8	**72.9**	67.1	**73.0**	67.2
Llama3-8B	FP16	6.14	9.45	80.9	77.7	53.2	79.2	72.9	81.2	74.2
	RTN	29.21	43.1	68.5	50.1	35.4	53.5	58.8	63.7	55.0
	NORMSUB (RTN)	15.1	22.0	73.2	57.5	35.4	63.2	63.3	67.0	59.9
	AWQ	9.53	14.74	76.1	69.2	42.2	71.4	69.0	78.2	67.7
	NORMSUB (AWQ)	8.97	14.09	77.8	71.0	44.5	71.6	70.1	76.3	68.6
	GPTQ	8.33	13.11	77.5	70.6	43.7	71.9	71.4	77.9	68.8
	NORMSUB (GPTQ)	7.87	12.13	78.1	72.3	44.2	74.2	72.3	**79.2**	70.1
	GPTAQ	8.13	12.77	77.3	69.3	43.6	61.9	72.0	77.7	67.0
	NORMSUB (GPTAQ)	7.85	12.28	77.3	69.7	44.6	67.8	**72.5**	78.1	68.3
	ResComp*	7.77	12.25	77.7	73.8	45.7	74.6	72.3	79.1	70.5
	NORMSUB (ResComp*)	**7.62**	**12.08**	**79.2**	**75.2**	**48.1**	**75.3**	72.4	77.2	**71.2**
Qwen3-8B	FP16	9.72	15.40	77.8	80.9	56.7	74.9	67.8	86.6	74.1
	RTN	23.54	33.86	68.2	58.7	35.3	54.6	54.9	63.9	55.9
	NORMSUB (RTN)	16.56	23.64	73.9	68.8	43.5	63.7	63.2	76.6	65.0
	AWQ	12.11	18.49	74.3	71.3	45.8	67.8	63.3	82.5	67.5
	NORMSUB (AWQ)	11.50	17.44	74.8	71.6	46.9	68.5	64.8	82.9	68.3
	GPTQ	11.29	17.09	76.9	76.8	50.9	71.6	69.2	**85.8**	71.9
	NORMSUB (GPTQ)	**10.51**	**16.15**	76.8	76.8	52.1	**72.6**	**69.4**	**85.8**	72.3
	GPTAQ	11.37	17.13	75.7	72.2	48.7	70.9	67.9	85.7	70.2
	NORMSUB (GPTAQ)	10.68	16.42	76.3	72.1	48.9	71.5	68.5	85.0	70.4
	ResComp*	11.66	17.45	76.6	77.6	53.2	70.7	67.3	84.7	71.7
	NORMSUB (ResComp*)	**10.51**	**16.15**	**77.5**	**78.3**	**54.6**	71.3	67.9	84.7	**72.4**

Table 5: **Post-only W3A16 recovery on MoE models (group size 128).** Highlighted rows apply NORMSUB-POST to GPTQ. Routers and KV caches remain unquantized. Avg. is the six-task mean; bold marks the best quantized result.

Model	Method	Perplexity ↓		Zero-shot Accuracy (%) ↑						Avg. ↑
		Wiki2	C4	PIQA	ARC-E	ARC-C	HS	WG	BoolQ	
OLMoE-1B-7B	FP16	6.65	12.14	79.7	76.9	48.5	78.2	69.7	70.3	70.6
	GPTQ	8.32	13.45	**78.3**	70.4	45.9	74.1	**69.4**	69.4	67.9
	GPTQ + NORMSUB-POST	**8.22**	**13.31**	78.1	**71.5**	**46.1**	**74.3**	68.7	**69.5**	**68.1**
Qwen3-30B-A3B	FP16	8.71	14.05	80.5	79.3	56.6	77.7	69.4	88.6	75.4
	GPTQ	10.33	15.86	77.7	72.3	53.6	74.2	**68.3**	87.7	72.3
	GPTQ + NORMSUB-POST	**9.19**	**14.47**	**78.3**	**73.6**	**54.6**	**75.1**	68.2	**87.9**	**73.0**

5 CONCLUSION

We study normalization-affine parameters as a structured correction subspace for quantization recovery. In a matched-budget fixed-graph comparison, affine updates outperform the tested equal-size weight controls. Functional diagnostics show that task-level recovery need not uniformly reduce local reconstruction error. NORMSUB combines backend-aligned proxy adaptation, affine-only transfer to a restored floating-point model for PTQ reconstruction, and refinement of the rebuilt graph's affine coordinates. The complete pipeline improves perplexity and mean zero-shot accuracy across the reported dense language-model configurations. MoE and post-QAT studies evaluate post-only adaptation, with checkpoint-dependent QAT outcomes. These results highlight the role of coordinate structure in budget-limited recovery. Scope and limitations are discussed in Appendix G.

REFERENCES

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L. Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. QuaRot: Outlier-free 4-bit inference in rotated LLMs. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/b5b939436789f76f08b9d0da5e81af7c-Abstract-Conference.html.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 1–9, 2022. doi: 10.18653/v1/2022.acl-short.1. URL https://aclanthology.org/2022.acl-short.1/.

Yash Bhalgat, Jinwon Lee, Markus Nagel, Tijmen Blankevoort, and Nojun Kwak. LSQ+: Improving low-bit quantization through learnable offsets and better initialization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 696–697, 2020. URL https://openaccess.thecvf.com/content_CVPRW_2020/html/w40/Bhalgat_LSQ_Improving_Low-Bit_Quantization_Through_Learnable_Offsets_and_Better_Initialization_CVPRW_2020_paper.html.

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. doi: 10.1609/aaai.v34i05.6239. URL https://ojs.aaai.org/index.php/AAAI/article/view/6239.

Tianyi Chen, Sihan Chen, Xiaoyi Qu, Dan Zhao, Ruomei Yan, Jongwoo Ko, Luming Liang, and Pashmina Cameron. StableQAT: Stable quantization-aware training at ultra-low bitwidths. *arXiv preprint arXiv:2601.19320*, 2026. URL https://github.com/microsoft/StableQAT.

Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. PACT: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018. URL https://arxiv.org/abs/1805.06085.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. 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)*, pp. 2924–2936, 2019. doi: 10.18653/v1/N19-1300. URL https://aclanthology.org/N19-1300/.

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. URL https://arxiv.org/abs/1803.05457.

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3213–3223, 2016. URL https://openaccess.thecvf.com/content_cvpr_2016/html/Cordts_The_Cityscapes_Dataset_CVPR_2016_paper.html.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. URL https://www.image-net.org/static_files/papers/imagenet_cvpr09.pdf.

Steven K. Esser, Jeffrey L. McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S. Modha. Learned step size quantization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020. URL https://openreview.net/pdf?id=rkgO66VKDS.

Jonathan Frankle, David J. Schwab, and Ari S. Morcos. Training batchnorm and only batchnorm: On the expressive power of random features in CNNs. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/pdf?id=vYeQQ29Tbvx.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=tcbBPnfwxS.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, 2023. URL https://zenodo.org/records/10256836.

Ruihao Gong, Xianglong Liu, Shenghu Jiang, Tianxiang Li, Peng Hu, Jiazhen Lin, Fengwei Yu, and Junjie Yan. Differentiable soft quantization: Bridging full-precision and low-bit neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4852–4861, 2019. URL https://openaccess.thecvf.com/ICCV2019.py?day=2019-10-31.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. URL https://arxiv.org/abs/1503.02531.

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 *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022. URL https://www.microsoft.com/en-us/research/?p=759361.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 448–456, 2015. URL https://proceedings.mlr.press/v37/ioffe15.html.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2704–2713, 2018. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Jacob_Quantization_and_Training_CVPR_2018_paper.html.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. URL https://cave.cs.toronto.edu/kriz/. Technical report.

Liang Li, Qingyuan Li, Bo Zhang, and Xiangxiang Chu. Norm tweaking: High-performance low-bit quantization of large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 18536–18544, 2024. doi: 10.1609/aaai.v38i17.29815. URL https://ojs.aaai.org/index.php/AAAI/article/view/29815.

Shuaiting Li, Juncan Deng, Kedong Xu, Rongtao Deng, Hong Gu, Minghan Jiang, Haibin Shen, and Kejie Huang. Rethinking residual errors in compensation-based LLM quantization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2026. URL https://openreview.net/forum?id=LWYZ1nNkJl.

Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. BRECQ: Pushing the limit of post-training quantization by block reconstruction. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021a. URL https://openreview.net/references/pdf?id=i8MSzRARRz.

Yuhang Li, Mingzhu Shen, Jian Ma, Yan Ren, Mingxin Zhao, Qi Zhang, Ruihao Gong, Fengwei Yu, and Junjie Yan. MQBench: Towards reproducible and deployable model quantization benchmark. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021b. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/hash/c20ad4d76fe97759aa27a0c99bff6710-Abstract-round1.html.

Yuhang Li, Ruokai Yin, Donghyun Lee, Shiting Xiao, and Priyadarshini Panda. GPTAQ: Efficient finetuning-free quantization for asymmetric calibration. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 36690–36706, 2025. URL https://proceedings.mlr.press/v267/li25dn.html.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. In *Proceedings of the Conference on Machine Learning and Systems (MLSys)*, volume 6, pp. 87–100, 2024. URL https://proceedings.mlsys.org/paper_files/paper/2024/hash/42a452cbafa9dd64e9ba4aa95cc1ef21-Abstract-Conference.html.

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. SpinQuant: LLM quantization with learned rotations. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/e5b1c0d4866f72393c522c8a00eed4eb-Abstract-Conference.html.

Llama team. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. URL https://ai.meta.com/research/publications/the-llama-3-herd-of-models/.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017. URL https://openreview.net/pdf?id=Byj72udxe. arXiv:1609.07843.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, Yuling Gu, Shane Arora, Akshita Bhagia, Dustin Schwenk, David Wadden, Alexander Wettig, Binyuan Hui, Tim Dettmers, Douwe Kiela, Ali Farhadi, Noah A. Smith, Pang Wei Koh, Amanpreet Singh, and Hannaneh Hajishirzi. OLMoE: Open mixture-of-experts language models. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/9b224ace8963c9385ad5e2b5c9039b97-Paper-Conference.pdf.

Markus Nagel, Rana Ali Amjad, Mart van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 7197–7206, 2020. URL https://proceedings.mlr.press/v119/nagel20a.html.

Markus Nagel, Marios Fournarakis, Yelysei Bondarenko, and Tijmen Blankevoort. Overcoming oscillations in quantization-aware training. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 16318–16330, 2022. URL https://proceedings.mlr.press/v162/nagel22a.html.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf.

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. URL https://jmlr.org/papers/v21/20-074.html.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pp. 234–241, 2015. doi: 10.1007/978-3-319-24574-4_28. URL https://doi.org/10.1007/978-3-319-24574-4_28.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8732–8740, 2020. doi: 10.1609/aaai.v34i05.6399. URL https://ojs.aaai.org/index.php/AAAI/article/view/6399.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. MobileNetV2: Inverted residuals and linear bottlenecks. In *Proceedings of*

the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4510–4520, 2018. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Sandler_MobileNetV2_Inverted_Residuals_CVPR_2018_paper.html.

Yuxuan Sun, Ruikang Liu, Haoli Bai, Han Bao, Kang Zhao, Yuening Li, Jiaxin Hu, Xianzhi Yu, Lu Hou, Chun Yuan, Xin Jiang, Wulong Liu, and Jun Yao. FlatQuant: Flatness matters for LLM quantization. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 57587–57613, 2025. URL https://proceedings.mlr.press/v267/sun25l.html.

Hugo Touvron, Louis Martin, Kevin Stone, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. URL https://arxiv.org/abs/2307.09288.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/pdf/4de0af9691a5dcc52de7de756676fded33d037ef.pdf.

Xiuying Wei, Ruihao Gong, Yuhang Li, Xianglong Liu, and Fengwei Yu. QDrop: Randomly dropping quantization for extremely low-bit post-training quantization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/pdf?id=ySQH0oDyp7.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, 2020. doi: 10.18653/v1/2020.emnlp-demos.6. URL https://aclanthology.org/2020.emnlp-demos.6/.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pp. 38087–38099, 2023. URL https://proceedings.mlr.press/v202/xiao23c.html.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://github.com/QwenLM/Qwen3#citation.

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. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472/.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/1e8a19426224ca89e83cef47f1e7f53b-Abstract.html.

Tianrui Zhu, Houyuan Chen, Ruihao Gong, Michele Magno, Haotong Qin, and Kai Zhang. Post-training quantization for video matting. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/65e876f6a98c6799d0b3145966dd73e2-Abstract-Conference.html.

A Reproducibility and Experimental Protocol

Dense LLM two-stage construction. For each backend, both PTQ constructions use the same fixed calibration tensor. The first completed quantized graph serves as the disposable backend-aligned proxy in Eq. (7a). After affine adaptation, we retain only $\mathbf{\Phi}_{\mathrm{pre}}$ and discard the proxy's quantized weights, quantizer state, and optimizer state. We restore the original floating-point checkpoint and transfer the affine state into the backend-prepared graph under its coordinate convention. The fresh construction $\mathcal{G}_2 = \mathcal{B}_{\mathcal{D}}(\mathcal{S}_0; \mathbf{\Phi}_{\mathrm{pre}})$ exposes $\mathbf{\Phi}_{\mathrm{init}} = \phi(\mathcal{G}_2)$, from which the post-stage starts with a fresh optimizer. Non-affine persistent state and quantization rules remain fixed during each affine stage, not during PTQ reconstruction.

Dense backend settings. The QuaRot-based backends use Hadamard rotations; GPTQ-family updates use damping 0.01 and activation ordering. W3A16 uses symmetric group-wise weight quantization with group size 128 and 128 C4 training sequences of 2,048 tokens. W2A4KV4 uses symmetric clipped W2 weights with group size 128, dynamic per-token asymmetric A4/K4/V4 quantization, and 128 WikiText-2 training sequences of 2,048 tokens. Calibration sampling uses seed 0.

Dense LLM affine optimization. Unless otherwise stated, each stage uses 200 AdamW updates, batch size 1, sequence length 256, learning rate 5×10^{-4}, zero weight decay, no learning-rate schedule, and gradient-norm clipping at 1.0. The objective combines equal-weight cross-entropy and full-vocabulary distillation with temperature $\tau = 1$. Both stages use the original floating-point model as the frozen teacher with `use_cache=False`. Only normalization-affine coordinates are optimized, although backpropagation through intermediate activations is required. Calibration subblocks are sampled deterministically with seed 100003, and checkpoints are saved every ten updates. For AWQ (Lin et al., 2024), relative log-scales use learning rate 10^{-4} and are clamped to $[-0.1, 0.1]$. The dense LLM experiments do not use the channel-scale regularizer.

Dense LLM evaluation. WikiText-2 perplexity uses the complete test split in non-overlapping 2,048-token blocks. C4 perplexity uses the first 1,100 validation documents and the first 256 full 2,048-token blocks. Zero-shot evaluation uses lm-eval-harness 0.4.9.1 (Gao et al., 2023), maximum length 2,048, and batch size 4. We report normalized accuracy for PIQA (Bisk et al., 2020), ARC-Easy and ARC-Challenge (Clark et al., 2018), and HellaSwag (Zellers et al., 2019), and accuracy for WinoGrande (Sakaguchi et al., 2020) and BoolQ (Clark et al., 2019). Avg. is their unweighted mean. The software stack uses PyTorch (Paszke et al., 2019), Transformers 4.54.1 (Wolf et al., 2020), Datasets 3.6.0, and Accelerate 1.9.0 on MetaX C500 and C600-A accelerators. Evaluation uses floating-point execution of quantize–dequantize graphs; the stated bit widths do not establish native low-bit speedups or packed-KV memory savings.

Baseline provenance. Paired comparisons are designed to keep the source checkpoint, tokenizer, dataset snapshot, calibration indices, preprocessing, and metric implementation fixed. The stage ablation reuses completed source runs for the baseline and first affine state. Its comparisons are made within Table 3; cross-table identities require matching run provenance.

Construction cost. Without reusing a compatible completed proxy, the full pipeline requires two PTQ constructions and two affine stages. Affine-only optimization reduces trainable parameters and optimizer state, but not all activation storage or model memory. End-to-end time and peak memory require measurements under matched conditions.

Stage ablation. Table 3 uses GPTQ for W3A16 and QuaRot+ResComp with GPTAQ for W2A4KV4. Within each setting, the four rows share the checkpoint, backend configuration, calibration tensor, and seed. PTQ performs no affine update. PTQ + NormSub-Post performs one 200-update refinement of the completed graph. NormSub-Pre + PTQ reconstruction transfers the first affine solution into a fresh backend-prepared floating-point graph and completes PTQ without post-adaptation. NormSub includes both separately optimized 200-update stages, each with a fresh optimizer. The budget is therefore matched per affine stage, not across complete runs.

Matched-budget parameter subspaces. The Qwen3-8B W2A4KV4 comparison starts from the same completed QuaRot–ResComp backbone and assigns each trainable branch $299{,}008$ scalar co-

ordinates. RandomW uses a fixed uniformly sampled weight mask. TopGradW ranks calibration-gradient magnitudes within a uniformly sampled candidate pool. AttentionW and MLPW restrict the same coordinate budget to attention and feed-forward projections, respectively. Masks are selected once and remain fixed. Weight-coordinate branches learn FP32 sparse increments over the frozen quantized operators; each increment receives the same processed input as its base operator and is cast to the input dtype in the forward pass. These increments are retained at evaluation without projection onto the low-bit weight grid. All trainable branches share the calibration blocks, task-plus-KD objective, 200 updates, batch size, sequence length, optimizer, and prescribed learning-rate settings. The Frozen branch performs no updates. This is the finite-budget coordinate comparison described in Section 3.2.

Mechanism probes. The four-state and structured-source measurements use 16 sequences of length 512 with token stride 16. The four-state intervention compares floating-point and quantized graphs in a common backend-compatible coordinate system, using their graph-local initial state $\mathbf{\Phi}_{\mathcal{G},0}$ and experimentally adapted state $\mathbf{\Phi}_{\mathcal{G},*}$. The adapted state is not the linearized optimum $\Delta\mathbf{\Phi}^*$ in Eq. (12). The reference is the original floating-point model. Identity-initialized post-fusion coordinates must not be interpreted as original normalization scales. The probes report common-reference logit NMSE, teacher KL, a teacher-Fisher quadratic form, true-token margin NMSE, and the full NLL gap. Structured-source interventions enable weight, activation, key-cache, and value-cache perturbations separately and jointly while keeping inputs, references, and affine states fixed. Recovery is reported as $1 - E(\mathbf{\Phi}_{\mathcal{G},*})/E(\mathbf{\Phi}_{\mathcal{G},0})$. A negative value denotes an increase in that discrepancy; it need not imply worse end-to-end performance.

MoE post-only experiments. We evaluate OLMoE (Muennighoff et al., 2025) and Qwen3 MoE (Yang et al., 2025). The MoE rows in Tables 5 and 7 apply NORMSUB-POST to completed quantized backbones. W3A16 uses GPTQ; the OLMoE W2A4 experiment uses the QuaRot-style MoE adapter with GPTQ. Router parameters and KV caches remain unquantized. These comparisons do not include pre-adaptation or a second PTQ construction.

SpinQuant extension. The learned-rotation initialization and fixed-rotation protocol for Table 10 are specified in Appendix E.

A.1 COMPLETE LANGUAGE-MODEL RESULTS

Tables 5 and 7 give the separate post-only MoE comparisons. All full-precision references and individual-task declines are retained.

Table 6: **W2A4KV4 results on dense language models (group size 128).** Highlighted rows apply the complete NORMSUB pipeline with the QuaRot-based backend in parentheses. The Avg. definition, boldface convention, and $\mathrm{ResComp}^*$ notation follow Table 4.

Model	Method	Perplexity ↓		Zero-shot Accuracy (%) ↑						Avg. ↑
		Wiki2	C4	PIQA	ARC-E	ARC-C	HS	WG	BoolQ	
Llama2-7B	FP16	5.47	7.27	78.9	74.6	46.1	75.9	69.2	77.7	70.4
	QuaRot + GPTAQ	11.7	24.8	62.3	45.6	25.6	41.2	54.0	62.4	48.5
	NORMSUB (QuaRot+GPTAQ)	8.2	**14.0**	66.8	50.0	**31.6**	**54.9**	58.2	64.5	54.3
	QuaRot + $\mathrm{ResComp}^*$	11.5	23.6	63.6	46.4	24.9	40.8	56.0	61.9	48.9
	NORMSUB (QuaRot+$\mathrm{ResComp}^*$)	**8.1**	14.2	**68.1**	**52.2**	30.7	54.8	**59.1**	**65.8**	**55.1**
Llama3-8B	FP16	6.14	9.45	80.9	77.7	53.2	79.2	72.9	81.2	74.2
	QuaRot + GPTAQ	23.0	62.8	55.8	37.2	22.3	36.5	50.4	59.3	43.6
	NORMSUB (QuaRot+GPTAQ)	**12.2**	**28.7**	63.9	**49.8**	29.3	**51.0**	57.2	64.9	52.7
	QuaRot + $\mathrm{ResComp}^*$	22.1	61.5	56.0	38.5	24.6	35.8	53.5	60.9	44.9
	NORMSUB (QuaRot+$\mathrm{ResComp}^*$)	**12.2**	29.1	**65.6**	49.7	**29.8**	50.7	**58.5**	**66.3**	**53.4**
Qwen3-8B	FP16	9.72	15.40	77.8	80.9	56.7	74.9	67.8	86.6	74.1
	QuaRot + GPTAQ	28.2	64.1	57.9	39.5	23.4	34.7	51.7	61.6	44.8
	NORMSUB (QuaRot+GPTAQ)	14.6	32.8	66.8	52.4	32.5	47.9	59.1	66.9	54.3
	QuaRot + $\mathrm{ResComp}^*$	18.7	38.3	61.6	44.4	28.6	43.7	54.8	64.4	49.6
	NORMSUB (QuaRot+$\mathrm{ResComp}^*$)	**12.0**	**25.4**	**69.1**	**56.8**	**37.1**	**53.6**	**62.1**	**70.1**	**58.1**

Table 7: **Post-only W2A4 recovery on OLMoE-1B-7B (group size 128).** The highlighted row applies NORMSUB-POST to the completed QuaRot+GPTQ graph built with the QuaRot-style MoE adapter. Router parameters and KV caches remain unquantized. Metric and boldface conventions follow Table 5.

Model	Method	Perplexity ↓		Zero-shot Accuracy (%) ↑						Avg. ↑
		Wiki2	C4	PIQA	ARC-E	ARC-C	HS	WG	BoolQ	
OLMoE-1B-7B	FP16	6.65	12.14	79.7	76.9	48.5	78.2	69.7	70.3	70.6
	QuaRot + GPTQ	17.81	22.83	69.8	53.9	29.7	48.9	57.1	**63.8**	53.9
	QuaRot + GPTQ + NORMSUB-POST	**15.79**	**20.15**	**71.4**	**54.7**	**31.1**	**50.7**	**59.3**	63.0	**55.0**

B Supplementary Mechanism Derivations and Limits

B.1 Exact BN Scale–Shift Cancellation

This derivation supplements the fixed-statistics BN example in Section 3.3. Assume fixed BN statistics (μ_c, σ_c) and a local pre-BN perturbation

$$\widehat{a}_c = \alpha_c a_c + \delta_c + e_c, \qquad \alpha_c > 0, \tag{8}$$

where e_c contains sample-dependent rounding, clipping, and saturation residuals. Let $\rho_c = \sqrt{\sigma_c^2 + \epsilon}$ and let (γ_c^0, β_c^0) denote the original BN affine parameters. The reference response is $y_c^0 = \gamma_c^0(a_c - \mu_c)/\rho_c + \beta_c^0$. Choosing

$$\gamma_c^* = \gamma_c^0/\alpha_c, \tag{9}$$

$$\beta_c^* = \beta_c^0 - \frac{\gamma_c^0}{\alpha_c \rho_c}\big[(\alpha_c - 1)\mu_c + \delta_c\big] \tag{10}$$

gives

$$\widehat{y}_c^* = y_c^0 + \frac{\gamma_c^0}{\alpha_c \rho_c} e_c, \qquad \|\widehat{y}_c^* - y_c^0\| \le \frac{|\gamma_c^0|}{\alpha_c \rho_c}\|e_c\|. \tag{11}$$

Thus, the structured scale–shift component is exactly cancelled under the stated assumptions, while the sample-dependent residual remains. This identity requires fixed BN statistics. RMSNorm has no bias term and provides multiplicative response control rather than exact cancellation of a general additive offset.

B.2 Weighted Projection and Its Limits

For the local response-matching problem in Eq. (6), a minimum-Euclidean-norm solution is

$$\Delta\mathbf{\Phi}^* = (\mathbf{J}_{\mathbf{\Phi}}^\top \mathbf{H}_z \mathbf{J}_{\mathbf{\Phi}})^\dagger \mathbf{J}_{\mathbf{\Phi}}^\top \mathbf{H}_z \boldsymbol{e}_z. \tag{12}$$

Here † denotes the Moore–Penrose pseudoinverse. Let $\boldsymbol{e}_\parallel = \mathbf{J}_{\mathbf{\Phi}}\Delta\mathbf{\Phi}^*$ and $\boldsymbol{e}_\perp = \boldsymbol{e}_z - \boldsymbol{e}_\parallel$. The residual is orthogonal to the locally attainable response directions under the $\mathbf{H}_z$ weighting. When $\mathbf{H}_z$ is singular, directions in its null space remain unpenalized. This is a first-order characterization of the surrogate: finite-step adaptation of the nonlinear model need not compute the exact projection. An uncorrected local residual does not by itself establish global flatness or irreversible loss of task information.

C Vision and QAT Extensions

This appendix supplements the complete vision pipelines and the post-only QAT checkpoint comparisons in Section 4.1.

C.1 One-Sided Affine Adaptation after QAT

Given a QAT checkpoint $(\mathbf{W}_{\mathrm{qat}}, \mathbf{\Phi}_{\mathrm{qat}}, \mathbf{\Omega}_{\mathrm{qat}})$, NormSub-Post optimizes the exposed normalization-affine coordinates using Eq. (3), while keeping all non-affine persistent state fixed, including backbone weights, quantizer state, and BN running statistics. This tests the recovery available from additional affine updates at that checkpoint, not convergence of the original QAT run.

Quantization graph. The CNN experiments use PyTorch (Paszke et al., 2019) and MQBench (Li et al., 2021b). Weights use symmetric per-output-channel quantization; activations use asymmetric per-tensor quantization with MSE observers and `FixedFakeQuantize`. ImageNet and Cityscapes quantize the full network. For CIFAR-100, the first and last quantized layers remain at 8 bits, while internal layers use the stated W/A precision. Strict fixed-graph controls reload an identical frozen observer state. ImageNet instead uses the epochwise observer-recalibration procedure below, so its quantizer state is not fixed across epochs.

Model	Method	W/A	Top-1 (%)	Δ (pp)
MobileNetV2	Full precision	32/32	71.30	–
	PACT	4/4	64.06	-7.24
	DSQ	4/4	67.36	-3.94
	LSQ	4/4	69.01	-2.29
	LSQ+	4/4	68.21	-3.09
	LSQ+ + NORMSUB-POST	4/4	69.83	-1.47
	OOQ-Dampen	4/4	66.21	-5.09
	OOQ-Dampen + NORMSUB-POST	4/4	68.07	-3.23
	OOQ-Freeze	4/4	70.02	-1.28
	OOQ-Freeze + NORMSUB-POST	4/4	**70.68**	**-0.62**
	StableQAT	4/4	68.24	-3.06
	StableQAT + NORMSUB-POST	4/4	69.12	-2.18
	PACT	2/4	57.90	-13.40
	DSQ	2/4	62.81	-8.49
	LSQ	2/4	64.58	-6.72
	LSQ+	2/4	63.42	-7.88
	LSQ+ + NORMSUB-POST	2/4	65.86	-5.44
	OOQ-Dampen	2/4	66.23	-5.07
	OOQ-Dampen + NORMSUB-POST	2/4	68.25	-3.05
	OOQ-Freeze	2/4	67.42	-3.88
	OOQ-Freeze + NORMSUB-POST	2/4	62.81	-8.49
	StableQAT	2/4	66.09	-5.21
	StableQAT + NORMSUB-POST	2/4	**68.50**	**-2.80**

Table 8: **Post-QAT affine adaptation on CIFAR-100 with MobileNetV2.** Highlighted rows apply NORMSUB-POST to the corresponding QAT checkpoints. We report top-1 accuracy (%); Δ is the signed difference from FP32 in percentage points. Bold marks the best quantized accuracy at each precision.

Some vision controls use the channel-scale regularizer

$$\mathcal{L}_{\mathrm{CS}} = \frac{1}{L}\sum_{\ell=1}^{L} \mathrm{Var}_c\big[\log\left(|\gamma_{\ell,c}| + \epsilon\right)\big], \tag{13}$$

where L is the number of regularized layers, Var_c is variance across channels, and $\epsilon > 0$ stabilizes the logarithm. The regularizer discourages a few channels from dominating the activation range. It is included only where specified below; the dense LLM experiments do not use it.

CIFAR-100 post-QAT adaptation. We evaluate MobileNetV2 (Sandler et al., 2018) on CIFAR-100 (Krizhevsky, 2009). The checkpoint comparisons include PACT (Choi et al., 2018), DSQ (Gong et al., 2019), LSQ (Esser et al., 2020), LSQ+ (Bhalgat et al., 2020), oscillation-aware methods (Nagel et al., 2022), and StableQAT (Chen et al., 2026). For each paired checkpoint, NORMSUB-POST freezes the convolutional, linear, and quantizer parameters and updates only BatchNorm scales and biases. Adaptation uses 40K images, with a fixed 4K subset for model selection. Trainable variants share the CE+KD objective, AdamW with learning rate 10^{-4} and zero weight decay, cosine annealing over 20 epochs, distillation temperature $\tau = 4$, KD weight 1, and gradient-norm clipping at 1.0. Paired runs share the original floating-point teacher and image order.

ImageNet and Cityscapes pipelines. The classification and segmentation experiments use ImageNet (Deng et al., 2009) and Cityscapes (Cordts et al., 2016), respectively; the segmentation model is U-Net (Ronneberger et al., 2015). NORMSUB (RTN) and NORMSUB (QDrop) both use pre-adaptation, restoration and affine-state transfer, PTQ reconstruction, and post-adaptation. RTN applies the calibrated fake-quantization graph without weight reconstruction. QDrop (Wei et al., 2022) performs block reconstruction using its released stochastic quantization-drop objective. Convolutional and linear weights remain frozen during affine adaptation; QDrop reconstruction occurs only in the backend stage. ImageNet uses top-1 evaluation at the standard validation resolution. Cityscapes follows the source U-Net checkpoint's crop, normalization, and validation procedure.

Paired rows share calibration images, preprocessing, fake-quantization modules, and evaluation code.

CNN optimization. ImageNet uses the full training set for affine adaptation with both backends. Its affine stages use AdamW with learning rate 5×10^{-4}, zero weight decay, batch size 128, distillation temperature $\tau = 4$, $\lambda_{\mathrm{KD}} = 1$, and gradient-norm clipping at 1.0. The coefficient of Eq. (13) is 10^{-4}. BN running statistics remain frozen. Quantizer calibration uses 20 mini-batches and at most 5K images; observers are recalibrated after each adaptation epoch. QDrop uses 32 calibration batches, 20K reconstruction iterations, drop probability 0.5, scale learning rate 4×10^{-5}, warm-up ratio 0.2, rounding weight 0.01, temperature range $[20, 2]$, and learned hard-sigmoid rounding. The CIFAR-100 source QAT stage uses 100 epochs of SGD with momentum 0.9, learning rate 10^{-3}, weight decay 5×10^{-4}, and cosine decay. Its NORMSUB-POST stage uses 20 epochs at learning rate 10^{-4}. Cityscapes uses 512×512 crops, batch size 4, and 128 fixed calibration images.

Scope. The ImageNet and Cityscapes results evaluate the complete stage ordering, with domain-specific data and observer protocols. The CIFAR-100 checkpoint comparisons instead isolate post-QAT affine adaptation.

D CONTROLS, DIAGNOSTICS, AND PRACTICAL BOUNDARIES

D.1 PARAMETER-SUBSPACE CONTROLS

Subset	Params.	W4A4	W3A3	Time
RTN	0	0.33	0.09	–
Random equal-size	34.2K	2.56	0.21	0.98×
Top-gradient equal-size	34.2K	12.48	2.67	1.07×
Top-Fisher equal-size	34.2K	18.35	4.92	1.22×
Classifier subset	34.2K	0.81	0.12	0.94×
NoNorm	3.47M	57.42	31.68	1.71×
Normalization-affine only	34.2K	**66.11**	**40.21**	1.00×
Full tuning	3.52M	64.28	38.74	1.83×

Table 9: **Parameter-subspace controls on ImageNet MobileNetV2.** Values are top-1 accuracy (%); bold marks the best value at each precision. Time is relative to the normalization-affine branch and includes parameter selection. Rows identify the updated parameter subset, not the number of stages.

E ROTATION-BACKEND EXTENSION

Table 10 evaluates the complete NORMSUB pipeline with model-specific SpinQuant rotations in place of QuaRot rotations. Each run starts from the released W16A4KV4 rotation checkpoint, then applies W2 weight quantization, the A4/KV4 graph, and the two affine stages described in Appendix A. Rotations remain fixed during affine adaptation. The corresponding manifests retain the rotation filename and checksum where available.

Table 10: **SpinQuant-based W2A4KV4 results.** Highlighted rows apply the complete NORMSUB pipeline with the baseline's fixed, model-specific learned rotation. Metric, boldface, and ResComp* conventions follow Table 4.

Model	Method	Perplexity ↓		Zero-shot Accuracy (%) ↑						Avg. ↑
		Wiki2	C4	PIQA	ARC-E	ARC-C	HS	WG	BoolQ	
Llama2-7B	FP16	5.47	7.27	78.9	74.6	46.1	75.9	69.2	77.7	70.4
	SpinQuant + GPTAQ	11.6	26.5	62.2	42.3	25.5	40.9	54.7	**63.4**	48.1
	NORMSUB (SpinQuant+GPTAQ)	8.6	**15.2**	65.6	**52.2**	**30.1**	53.5	**57.6**	63.1	**53.7**
	SpinQuant + ResComp*	11.1	24.7	62.7	45.4	27.6	42.3	55.0	62.3	49.2
	NORMSUB (SpinQuant+ResComp*)	**8.5**	15.3	**67.1**	50.9	27.9	**54.0**	54.7	63.2	53.0
Llama3-8B	FP16	6.14	9.45	80.9	77.7	53.2	79.2	72.9	81.2	74.2
	SpinQuant + GPTAQ	18.3	55.6	61.4	42.3	26.6	40.1	54.7	62.7	47.9
	NORMSUB (SpinQuant+GPTAQ)	14.0	**32.6**	**63.6**	47.1	27.8	**50.4**	**56.8**	**65.9**	**51.9**
	SpinQuant + ResComp*	18.1	53.7	60.6	41.4	28.1	40.8	54.1	62.1	47.9
	NORMSUB (SpinQuant+ResComp*)	**13.1**	34.3	61.3	**47.2**	**28.9**	49.7	55.7	64.3	51.2

F ONE-SIDED POST-QUANTIZATION CORRECTION

Tables 11 and 12 evaluate NORMSUB-POST on completed dense-LLM PTQ graphs. Each comparison keeps non-affine persistent state fixed and refines the exposed normalization-affine coordinates from their graph-local initialization. These component results complement the complete-pipeline comparisons in Tables 4 and 6.

Table 11: **Post-only W3A16 recovery on dense language models (group size 128).** Highlighted rows apply NORMSUB-POST to the corresponding completed PTQ graph, keeping non-affine persistent state fixed. Metric, boldface, and ResComp* conventions follow Table 4.

Model	Method	Perplexity ↓		Zero-shot Accuracy (%) ↑						Avg. ↑
		Wiki2	C4	PIQA	ARC-E	ARC-C	HS	WG	BoolQ	
Llama2-7B	FP16	5.47	7.27	78.9	74.6	46.1	75.9	69.2	77.7	70.4
	RTN	8.22	11.53	76.3	66.5	40.6	68.7	65.8	67.5	64.2
	RTN + NORMSUB-POST	7.47	10.42	77.3	68.9	41.7	70.9	66.0	70.8	65.9
	AWQ	6.79	8.93	77.1	69.9	41.8	71.2	**67.7**	71.5	66.3
	AWQ + NORMSUB-POST	6.53	8.71	77.2	**70.7**	**43.0**	72.4	67.2	72.8	**67.2**
	GPTQ	6.75	13.72	76.8	66.5	39.7	68.3	67.5	68.9	64.6
	GPTQ + NORMSUB-POST	6.31	8.33	77.2	66.3	41.6	72.2	67.6	69.3	65.7
	GPTAQ	6.81	8.40	77.8	69.2	40.7	71.9	67.6	71.5	66.5
	GPTAQ + NORMSUB-POST	6.31	8.18	**78.2**	70.5	40.9	72.0	66.6	71.2	66.6
	ResComp*	6.25	8.19	77.4	69.5	41.5	72.3	67.4	71.5	66.6
	ResComp* + NORMSUB-POST	**6.22**	**8.16**	77.5	69.4	42.3	**72.7**	67.0	**73.2**	67.0
Llama3-8B	FP16	6.14	9.45	80.9	77.7	53.2	79.2	72.9	81.2	74.2
	RTN	29.21	43.1	68.5	50.1	35.4	53.5	58.8	63.7	55.0
	RTN + NORMSUB-POST	15.2	22.3	73.1	57.1	35.5	63.2	63.2	66.9	59.8
	AWQ	9.53	14.74	76.1	69.2	42.2	71.4	69.0	78.2	67.7
	AWQ + NORMSUB-POST	9.09	14.10	77.6	71.1	44.2	71.4	70.1	76.5	68.5
	GPTQ	8.33	13.11	77.5	70.6	43.7	71.9	71.4	77.9	68.8
	GPTQ + NORMSUB-POST	7.89	12.21	78.2	72.1	44.5	74.4	72.3	**79.1**	70.1
	GPTAQ	8.13	12.77	77.3	69.3	43.6	61.9	72.0	77.7	67.0
	GPTAQ + NORMSUB-POST	7.97	12.36	77.2	69.8	44.3	67.6	72.4	78.1	68.2
	ResComp*	7.77	12.25	77.7	73.8	45.7	74.6	72.3	**79.1**	70.5
	ResComp* + NORMSUB-POST	**7.65**	**12.17**	**79.2**	**75.1**	**47.9**	**75.1**	**72.5**	77.1	**71.2**
Qwen3-8B	FP16	9.72	15.40	77.8	80.9	56.7	74.9	67.8	86.6	74.1
	RTN	23.54	33.86	68.2	58.7	35.3	54.6	54.9	63.9	55.9
	RTN + NORMSUB-POST	17.06	24.86	73.0	67.3	43.2	63.7	62.1	76.4	64.3
	AWQ	12.11	18.49	74.3	71.3	45.8	67.8	63.3	82.5	67.5
	AWQ + NORMSUB-POST	11.53	17.64	74.8	71.8	46.9	68.4	63.9	82.8	68.1
	GPTQ	11.29	17.09	76.9	76.8	50.9	71.6	**69.2**	**85.8**	71.9
	GPTQ + NORMSUB-POST	**10.59**	**16.38**	76.8	76.9	51.9	**72.4**	**69.2**	**85.8**	72.2
	GPTAQ	11.37	17.13	75.7	72.2	48.7	70.9	67.9	85.7	70.2
	GPTAQ + NORMSUB-POST	10.88	16.61	75.9	71.3	48.9	71.3	68.7	85.2	70.2
	ResComp*	11.66	17.45	76.6	77.6	53.2	70.7	67.3	84.7	71.7
	ResComp* + NORMSUB-POST	11.02	16.78	**77.3**	**78.2**	**54.1**	71.4	67.8	84.8	**72.3**

Table 12: **Post-only W2A4KV4 recovery on dense language models (group size 128).** Highlighted rows apply NORMSUB-POST to the corresponding completed QuaRot-based graph, keeping non-affine persistent state fixed. Metric, boldface, and $\mathrm{ResComp}^*$ conventions follow Table 4.

Model	Method	Perplexity ↓		Zero-shot Accuracy (%) ↑						Avg. ↑
		Wiki2	C4	PIQA	ARC-E	ARC-C	HS	WG	BoolQ	
Llama2-7B	FP16	5.47	7.27	78.9	74.6	46.1	75.9	69.2	77.7	70.4
	QuaRot + GPTAQ	11.7	24.8	62.3	45.6	25.6	41.2	54.0	62.4	48.5
	QuaRot + GPTAQ + NORMSUB-POST	8.4	**14.6**	66.2	49.5	**30.5**	**54.7**	57.5	63.3	53.6
	QuaRot + $\mathrm{ResComp}^*$	11.5	23.6	63.6	46.4	24.9	40.8	56.0	61.9	48.9
	QuaRot + $\mathrm{ResComp}^*$ + NORMSUB-POST	**8.3**	14.7	**66.6**	**50.2**	29.1	54.3	**58.2**	**64.9**	**53.9**
Llama3-8B	FP16	6.14	9.45	80.9	77.7	53.2	79.2	72.9	81.2	74.2
	QuaRot + GPTAQ	23.0	62.8	55.8	37.2	22.3	36.5	50.4	59.3	43.6
	QuaRot + GPTAQ + NORMSUB-POST	12.9	**29.9**	63.2	**49.2**	**28.7**	**50.8**	56.9	64.7	52.3
	QuaRot + $\mathrm{ResComp}^*$	22.1	61.5	56.0	38.5	24.6	35.8	53.5	60.9	44.9
	QuaRot + $\mathrm{ResComp}^*$ + NORMSUB-POST	**12.8**	30.3	**64.7**	48.4	28.3	50.2	**57.7**	**65.6**	**52.5**
Qwen3-8B	FP16	9.72	15.40	77.8	80.9	56.7	74.9	67.8	86.6	74.1
	QuaRot + GPTAQ	28.2	64.1	57.9	39.5	23.4	34.7	51.7	61.6	44.8
	QuaRot + GPTAQ + NORMSUB-POST	15.0	33.5	65.5	51.4	29.5	46.3	57.1	66.6	52.7
	QuaRot + $\mathrm{ResComp}^*$	18.7	38.3	61.6	44.4	28.6	43.7	54.8	64.4	49.6
	QuaRot + $\mathrm{ResComp}^*$ + NORMSUB-POST	**12.7**	**26.8**	**66.9**	**56.1**	**34.5**	**52.5**	**59.2**	**69.7**	**56.5**

G DISCUSSION AND LIMITATIONS

Scope of the evidence. The controlled coordinate study and functional diagnostics support normalization-affine correction under the tested adaptation budgets. They do not establish that finite-step adaptation computes an exact projection or uniformly reduces local reconstruction error. The complete language-model pipeline is evaluated on dense 7B–8B models, whereas the MoE experiments evaluate post-only adaptation.

Adaptation cost and uncertainty. The full pipeline includes two affine-adaptation stages and a second PTQ construction. The stage ablation matches the update budget per affine stage, not the total computation across complete runs. The experiments use a single deterministic seed; multi-seed uncertainty and compute-matched comparisons with longer post-only adaptation remain untested.

Inference scope. The reported quantize–dequantize evaluations measure model accuracy rather than native low-bit execution performance. They do not establish low-bit speedups, packed-KV memory savings, or deployment latency. No learned auxiliary branch is added at inference; backend-specific fusion equivalence requires separate verification.

LARGE LANGUAGE MODEL (LLM) USAGE

LLM tools assisted with drafting, language revision, and code development. The authors are responsible for experimental design, verification of results, and all claims in the manuscript.