BITMOE: POST-TRAINING BINARIZATION FOR MOE-BASED LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) with Mixture-of-Experts (MoE) architectures have achieved remarkable progress, yet their substantial weight storage and memory traffic hinder practical deployment. Weight binarization offers extreme compression, but transferring dense-model methods to MoE faces three coupled challenges: decomposition–binarization mismatch, interaction-dependent residual allocation, and quantization-induced expert-shift. To address these challenges, we propose **BitMoE**, a post-training binarization framework coordinating shared representation, residual capacity, and routing stability. First, Routing-Coupled Binary Factorization alternately refines higher-precision shared factors and expert-specific binary factors to reduce the reconstruction error of routed expert aggregation. Second, Conditional Expert Residual Allocation selects additional binary residual blocks by their conditional reconstruction gain per added storage bit after readapting the shared output factor, updating gains as corrections accumulate. Third, Margin-Aware Optimal Brain Routing uses boundary-aware curvature to preserve selected–unselected expert margins and mitigate expert-shift. BitMoE retains all experts and performs optimization offline under an explicit storage budget. Extensive experiments demonstrate that BitMoE consistently outperforms state-of-the-art binary baselines across multiple MoE models and benchmarks. For instance, on Qwen3-30B-A3B, BitMoE delivers 54.2% lower perplexity and improves average zero-shot accuracy by 19.71 percentage points over the strongest binary baseline, alongside an over 2× inference speedup over the BF16 model.

1 INTRODUCTION

Mixture-of-Experts (MoE) has emerged as a powerful architecture for scaling large language models (LLMs) (DeepSeek-AI et al., 2025; Yang et al., 2025; Muennighoff et al., 2025). By combining specialized experts with a learned router that activates only a few experts per token, MoE-based LLMs expand model capacity without proportionally increasing computation. However, sparse activation does not eliminate weight storage: in an all-experts-resident deployment, inactive experts still occupy memory. For example, GPT-OSS-120B (OpenAI et al., 2025) contains 117 billion parameters but activates only 5.1 billion per token; storing all its weights in BF16 would require approximately 234 GB, excluding runtime states. The resulting storage and weight-transfer costs motivate extreme weight compression, but preserving model quality under binarization remains challenging.

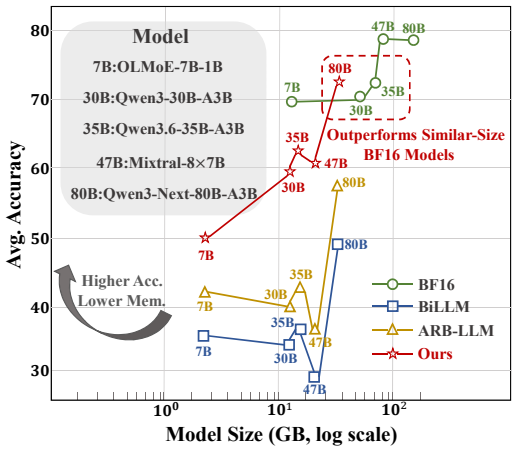


Figure 1: Average 7-task zero-shot accuracy across MoE LLMs and binarization methods.

Post-Training Quantization (PTQ) offers an effective route to reducing LLM storage and data-movement costs without expensive retraining. Existing binarization methods, including BiLLM (Huang et al., 2024) and ARB-LLM (Li et al., 2024), use importance-guided reconstruc-

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.

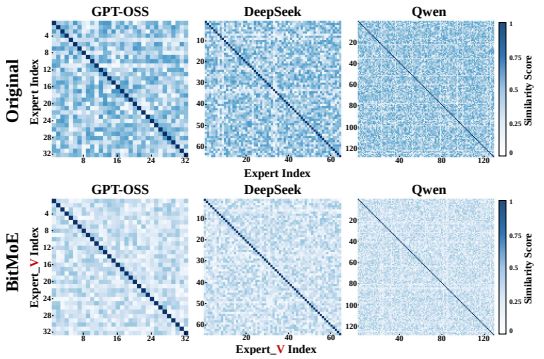


Figure 2: Centered Kernel Alignment (CKA) similarity across experts in GPT-OSS-20B, DeepSeek-V2-Lite, and Qwen3-30B-A3B. **Top**: original weights. **Bottom**: expert-specific V factors after joint SVD.

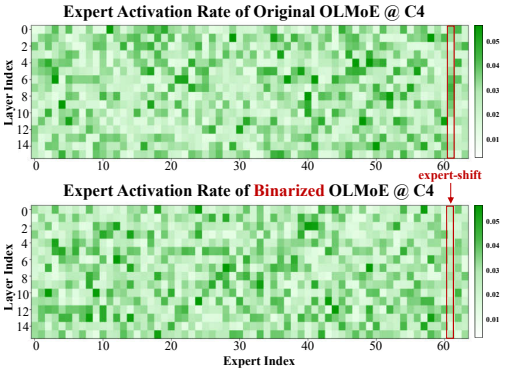


Figure 3: Token–expert mappings of OLMoE on C4 (**Top**: original model; **Bottom**: binarized model). Changes in token assignments illustrate quantization-induced expert-shift relative to the original model.

tion and binary residuals to protect selected weights, while other approaches represent weights as products of scaled binary matrices (Boža & Macko, 2025). Related work also explores shared expert decomposition for merging (Li et al., 2025) and end-loss-guided quantization (Kim et al., 2025). Yet MoE outputs are formed through input-dependent expert aggregation, where reconstruction errors can reinforce or offset one another. Optimizing experts or allocating residuals independently does not explicitly account for this joint behavior. The degradation from directly applying dense-model binarization (Fig. 1) motivates coordinating binary representations, residual capacity, and routing.

To understand these difficulties, we examine MoE binarization from the perspectives of shared representation, residual allocation, and routing stability. ❶ **Decomposition–binarization mismatch.** Cross-expert similarity motivates sharing (Fig. 2), but a basis fitted to full-precision weights may be poorly matched to binary expert factors. Fixing this basis before binarization prevents it from adapting to the resulting quantization errors. ❷ **Interaction-dependent residual allocation.** The value of an additional binary residual depends on the corrections already selected, the errors of co-activated experts, and the compensation available through shared-factor readaptation. Static, independent importance scores can therefore misallocate a limited residual budget, even when they capture individual weight sensitivity. ❸ **Quantization-induced expert-shift.** Quantization can change Top-k expert assignments (Fig. 3), altering the expert combinations used during reconstruction. Such changes can undermine both aggregate reconstruction and residual allocation, making routing preservation integral to the compression process.

To this end, we propose **BitMoE**, a post-training binarization framework that coordinates *shared representation, residual capacity, and routing stability* while retaining all experts. ❶ **Routing-Coupled Binary Factorization (RCBF)** uses joint SVD for initialization, then alternately refines higher-precision shared factors and expert-specific binary factors to reduce the reconstruction error of routed expert aggregation. ❷ **Conditional Expert Residual Allocation (CERA)** builds on this representation to select additional binary residual blocks by their conditional reconstruction benefit per added storage bit after readapting the existing shared output factor, refreshing these benefits as corrections accumulate. ❸ **Margin-Aware Optimal Brain Routing (MOBR)** uses boundary-aware curvature to protect the margins between selected and unselected experts, mitigating expert-shift and stabilizing the routing conditions used by RCBF and CERA. Together, these components adapt the base binary representation, allocate residual capacity, and protect expert selection, with all optimization performed offline. Extensive experiments demonstrate that BitMoE outperforms existing binary PTQ baselines across diverse MoE-based LLMs. On Qwen3-30B-A3B, it reduces perplexity by more than 50% relative to ARB-LLM and achieves over 2× inference speedup over the BF16 model under our evaluation protocol. Our key contributions can be summarized as follows:

- We formulate MoE binarization around three challenges: decomposition–binarization mismatch, interaction-dependent residual allocation, and expert-shift.
- We introduce **BitMoE**, integrating routing-coupled binary factorization, conditional residual allocation with shared-factor readaptation, and margin-aware routing preservation within a common deployment-oriented framework.

- Extensive experiments across diverse MoE-based LLM architectures demonstrate that BitMoE consistently outperforms state-of-the-art binary PTQ methods while achieving favorable compression–accuracy trade-offs and practical inference efficiency.

2 Related Work

Mixture-of-Experts LLMs and Compression. The MoE paradigm (Jacobs et al., 1991; Jordan & Jacobs, 1994) combines expert specialization with input-dependent sparse routing. In modern LLMs, Switch Transformers (Fedus et al., 2022) adopt top-1 routing, Mixtral-8x7B (Jiang et al., 2024) activates two experts, while the DeepSeek series (Dai et al., 2024; DeepSeek-AI et al., 2024; 2025) and Qwen1.5-MoE (Team, 2024b) combine shared and routed experts. Kimi-K2 (Team et al., 2025) further scales MoE to one trillion parameters with 32B activated per token. The resulting parameter scale has motivated substantial work on MoE compression. Sub-MoE (Li et al., 2025) performs joint SVD within expert clusters and merges expert-specific factors, while KBVQ-MoE (Xu et al., 2026) combines input-guided shared decomposition with vector quantization and output-bias correction. MxMoE (Duanmu et al., 2025) further exploits heterogeneous expert sensitivity and activation frequency for hardware-aware mixed-precision allocation. These methods effectively reduce MoE deployment costs, but mainly focus on expert reduction, vector quantization, or multi-bit precision allocation; under extreme binary compression, jointly adapting shared expert representations to quantization error while preserving routing behavior remains challenging.

Binarization for Large Language Models. Weight binarization provides extreme compression by representing weights with binary codes and compact scaling parameters. BinaryBERT (Bai et al., 2021) explores binary Transformer models through weight splitting and fine-tuning, while PB-LLM (Shang et al., 2023) preserves salient weights at higher precision. BiLLM (Huang et al., 2024) introduces residual binarization for salient weights, and ARB-LLM (Li et al., 2024) improves binary reconstruction through alternating refinement. Other work combines structured sparsity with binarization (Dong et al., 2024) or factorizes weights into products of scaled binary matrices (Boža & Macko, 2025). GuidedQuant (Kim et al., 2025) further incorporates end-loss gradients into quantization objectives to better reflect downstream sensitivity. Despite these advances, most binary PTQ methods are developed for dense LLMs and optimize individual weight matrices or their local corrections. In MoE models, however, errors from co-activated experts interact after routing, the value of residual corrections depends on the current quantized representation, and router perturbations can change the expert set itself. These coupled effects make extreme MoE binarization fundamentally different from independently quantizing dense Transformer layers.

3 Method

Notation and setup. Consider a MoE layer with E experts $\{f_e\}_{e=1}^{E}$. For input $\boldsymbol{x} \in \mathbb{R}^d$, the router selects $S_k(\boldsymbol{x})$ with aggregation weights $p_e(\boldsymbol{x})$, giving the routed output

$$\mathrm{MoE}_{\mathrm{routed}}(\boldsymbol{x}) = \sum_{e=1}^{E} p_e(\boldsymbol{x}) f_e(\boldsymbol{x}), \qquad p_e(\boldsymbol{x}) = 0 \text{ for } e \notin S_k(\boldsymbol{x}). \tag{1}$$

Here f_e denotes the nonlinear expert; always-active shared experts, when present, contribute separately. We retain each model's routing normalization. For $\boldsymbol{W} \in \mathbb{R}^{n\times m}$, $\boldsymbol{B} \in \{-1,+1\}^{n\times m}$ denotes binary codes and $\widehat{\boldsymbol{W}}$ their scaled reconstruction; a basic approximation is $\widehat{\boldsymbol{W}} = \alpha\boldsymbol{B}$. BitMoE binarizes expert-specific factors, retains higher-precision shared factors, and allocates additional binary residuals under a storage budget that includes all retained scales and selection metadata.

Overview. To address the three challenges of MoE binarization identified in Sec. 1, we propose BitMoE, a three-component framework (Fig. 4). First, to resolve decomposition–binarization mismatch, Routing-Coupled Binary Factorization (RCBF) jointly refines higher-precision shared factors and binary expert factors against routed aggregate reconstruction (Sec. 3.1). Second, to capture interaction-dependent residual utility, Conditional Expert Residual Allocation (CERA) allocates additional binary residuals according to their conditional reconstruction gains under a fixed storage budget (Sec. 3.2). Third, to mitigate expert-shift, Margin-Aware Optimal Brain Routing

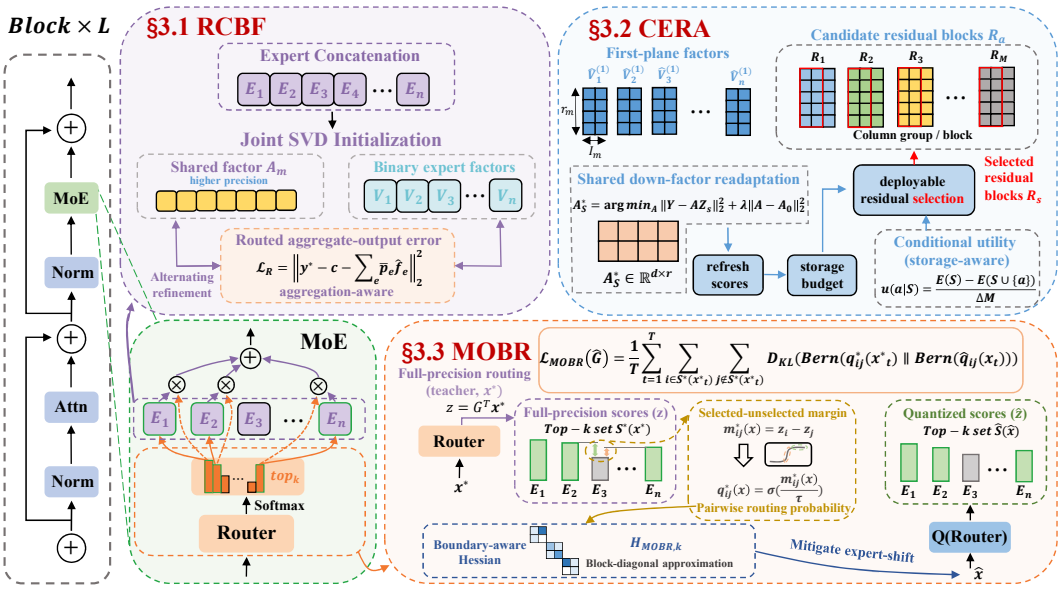


Figure 4: BitMoE couples **RCBF** for binary-aware shared-factor refinement, **CERA** for conditional residual allocation, and **MOBR** for routing-margin preservation. During layerwise calibration, MOBR fixes the deployed routing before RCBF and CERA optimize the expert representations.

(MOBR) preserves selected–unselected routing margins through boundary-aware router quantization (Sec. 3.3). The complete workflow and pseudocode are provided in Appendix A.3.

3.1 Routing-Coupled Binary Factorization (RCBF)

Motivation. Cross-expert similarity (Fig. 2) suggests that expert weights can share a compact representation. However, a decomposition optimized for full-precision weights is not necessarily well aligned with the binary factors used at deployment. Once the expert-specific components are binarized, their reconstruction errors are further combined through sparse routing, so fixing the shared basis before quantization can leave it poorly adapted to the resulting aggregate error.

We propose **Routing-Coupled Binary Factorization (RCBF)**, which uses joint SVD only for initialization and subsequently adapts the shared and binary expert factors to routed aggregate reconstruction. For projection $m \in \{\text{gate}, \text{up}, \text{down}\}$, let $\boldsymbol{W}_e^{(m)} \in \mathbb{R}^{O_m \times I_m}$ denote the weight of expert e. RCBF first jointly decomposes the expert weights as

$$\left[\boldsymbol{W}_1^{(m)}, \ldots, \boldsymbol{W}_E^{(m)}\right] \approx \boldsymbol{A}_m \left[\boldsymbol{V}_1^{(m)}, \ldots, \boldsymbol{V}_E^{(m)}\right], \tag{2}$$

where $\boldsymbol{A}_m \in \mathbb{R}^{O_m \times r_m}$ is shared across experts and $\boldsymbol{V}_e^{(m)} \in \mathbb{R}^{r_m \times I_m}$ is expert-specific. We initialize $\boldsymbol{A}_m = \boldsymbol{U}_{m,r_m} \boldsymbol{\Sigma}_{m,r_m}$ from the rank-r_m truncated SVD and obtain $\boldsymbol{V}_e^{(m)}$ from blocks of the retained right singular factor. The decomposition is performed independently for each projection, preserving the nonlinear expert computation. Under binary deployment, RCBF represents each expert projection as

$$\widehat{\boldsymbol{W}}_e^{(m)} = \boldsymbol{A}_m \widehat{\boldsymbol{V}}_e^{(m,1)}, \qquad \widehat{\boldsymbol{V}}_e^{(m,1)} = \operatorname{diag}(\boldsymbol{s}_{e,m}^r) \boldsymbol{B}_{e,m} \operatorname{diag}(\boldsymbol{s}_{e,m}^c), \tag{3}$$

where $\boldsymbol{B}_{e,m} \in \{-1, +1\}^{r_m \times I_m}$, while $\boldsymbol{A}_m$ is retained at higher precision. Ranks and deployment precisions remain fixed during calibration, and the row/column scaling follows established binary reconstruction (Li et al., 2024). All retained shared factors and scales are included in the storage accounting. Rather than reconstructing each expert independently, RCBF directly optimizes the routed MoE output. Let $\bar{p}_{t,e}$ denote the deployed routing weight for calibration token t, $\widehat{\boldsymbol{x}}_t$ the quantized-prefix input, and $\boldsymbol{y}_t^\star$ the corresponding teacher output. Collecting the shared factors, binary codes, and scales into Θ, we minimize

$$\mathcal{L}_{\mathrm{R}}(\Theta) = \frac{1}{T} \sum_{t=1}^{T} \left\| \boldsymbol{y}_t^\star - \boldsymbol{c}_t - \sum_{e=1}^{E} \bar{p}_{t,e} \widehat{f}_e(\widehat{\boldsymbol{x}}_t; \Theta) \right\|_2^2, \tag{4}$$

where $\boldsymbol{c}_t$ contains fixed always-active expert contributions and $\widehat{f}_e$ denotes the complete nonlinear expert computation. This objective explicitly captures interactions among simultaneously activated experts. For expert-output perturbations $\boldsymbol{\delta}_{t,e}$,

$$\left\| \sum_e \bar{p}_{t,e} \boldsymbol{\delta}_{t,e} \right\|_2^2 = \sum_e \bar{p}_{t,e}^2 \|\boldsymbol{\delta}_{t,e}\|_2^2 + 2 \sum_{e<f} \bar{p}_{t,e} \bar{p}_{t,f} \langle \boldsymbol{\delta}_{t,e}, \boldsymbol{\delta}_{t,f} \rangle. \tag{5}$$

The cross terms capture reinforcement or compensation among co-activated expert errors, which independent reconstruction ignores. RCBF alternates shared-factor/scale refinement with binary-factor updates. Each update is evaluated using the aggregate reconstruction objective, and shared-factor candidates are requantized before acceptance. For the down projection, fixed intermediate activations yield a regularized least-squares update, whereas gate/up projections require differentiation through the nonlinear expert. Only loss-decreasing updates are committed. RCBF uses one binary plane for the base representation, while CERA allocates additional residual capacity. Detailed initialization, updates, and the least-squares derivation are provided in Appendix A.1.1.

3.2 Conditional Expert Residual Allocation (CERA)

Motivation. After RCBF, all expert projections share a compact base representation, yet their remaining binary reconstruction errors are highly heterogeneous. Allocating additional residual capacity using a fixed importance ranking is suboptimal because the value of a correction depends on which residuals have already been selected, how co-activated expert errors interact, and how much of the remaining error can be absorbed by shared-factor readaptation.

We propose **Conditional Expert Residual Allocation (CERA)**, which allocates additional binary residuals according to their conditional reconstruction benefit under a fixed storage budget. For each expert e and projection $m \in \{\text{gate}, \text{up}, \text{down}\}$, CERA fits a second binary plane to the residual left by the RCBF base representation and partitions it into disjoint column-group candidates $a = (e, m, g)$. Their binary codes and scales are fixed during allocation. For a selected candidate set $\mathcal{S}$, the deployed expert factor becomes

$$\widehat{\boldsymbol{V}}_e^{(m)}(\mathcal{S}) = \widehat{\boldsymbol{V}}_e^{(m,1)} + \sum_{a \in \mathcal{S}_{e,m}} \boldsymbol{R}_a, \tag{6}$$

where $\boldsymbol{R}_a$ is nonzero only on its associated column group. All gate/up/down candidates share the layer budget and per-expert capacities; first-plane factors and ranks remain fixed. A residual correction should be evaluated after accounting for the compensation available from the shared representation. For a selected set $\mathcal{S}$, we define

$$\boldsymbol{z}_t(\mathcal{S}) = \sum_e \bar{p}_{t,e} \widehat{\boldsymbol{V}}_e^{(\text{down})}(\mathcal{S}) \boldsymbol{h}_{t,e}(\mathcal{S}), \tag{7}$$

where $\boldsymbol{h}_{t,e}(\mathcal{S})$ is obtained through the complete gate/up nonlinear computation. Let $\boldsymbol{Z}_{\mathcal{S}}$ stack these latent aggregates and $\boldsymbol{Y}$ denote the corresponding teacher outputs after subtracting fixed contributions. Starting from the RCBF shared down factor $\boldsymbol{A}_0$, CERA readapts it by

$$\boldsymbol{A}_{\mathcal{S}}^{\star} = \arg\min_{\boldsymbol{A}} \|\boldsymbol{Y} - \boldsymbol{A}\boldsymbol{Z}_{\mathcal{S}}\|_F^2 + \lambda \|\boldsymbol{A} - \boldsymbol{A}_0\|_F^2 = (\boldsymbol{Y}\boldsymbol{Z}_{\mathcal{S}}^\top + \lambda \boldsymbol{A}_0)(\boldsymbol{Z}_{\mathcal{S}}\boldsymbol{Z}_{\mathcal{S}}^\top + \lambda \boldsymbol{I})^{-1}. \tag{8}$$

This readaptation modifies only the existing shared down factor and introduces no additional high-precision matrix. Gate/up candidates instead require recomputing their nonlinear responses. The continuous solution is requantized to its deployment precision before evaluation. CERA then evaluates each candidate by its conditional reconstruction gain per additional storage bit. Let $\mathcal{E}(\mathcal{S}; \boldsymbol{A}) = \|\boldsymbol{Y} - \boldsymbol{A}\boldsymbol{Z}_{\mathcal{S}}\|_F^2$ and $\boldsymbol{A}_{\mathcal{S}}$ denote the current deployed shared factor. For candidate a, let $\boldsymbol{A}_a$ be the better deployed choice between $\boldsymbol{A}_{\mathcal{S}}$ and the requantized $\boldsymbol{A}_{\mathcal{S}\cup\{a\}}^{\star}$. Its utility is

$$u(a \mid \mathcal{S}) = \frac{\mathcal{E}(\mathcal{S}; \boldsymbol{A}_{\mathcal{S}}) - \mathcal{E}(\mathcal{S} \cup \{a\}; \boldsymbol{A}_a)}{\mathcal{M}(\mathcal{S} \cup \{a\}) - \mathcal{M}(\mathcal{S})}, \tag{9}$$

where $\mathcal{M}$ counts occupied packed storage, including scales and metadata but excluding unused record capacity. Thus, the same residual block can have different value as the selected set evolves. With the shared factor fixed, the finite gain of a candidate can be written as

$$\Delta_{\text{fixed}}(a \mid \mathcal{S}) = 2\langle \boldsymbol{E}_{\mathcal{S}}, \boldsymbol{D}_a \rangle - \|\boldsymbol{D}_a\|_F^2, \tag{10}$$

where $\boldsymbol{E}_{\mathcal{S}} = \boldsymbol{Y} - \boldsymbol{A}_{\mathcal{S}}\boldsymbol{Z}_{\mathcal{S}}$ is the current residual and $\boldsymbol{D}_a = \boldsymbol{A}_{\mathcal{S}}(\boldsymbol{Z}_{\mathcal{S}\cup\{a\}} - \boldsymbol{Z}_{\mathcal{S}})$ is the candidate-induced output change. Hence, a large correction is useful only when it aligns with the remaining reconstruction error. CERA commits the highest-utility feasible shortlist candidate with gain above tolerance, refreshes scores, and stops when no improving candidate remains or the evaluation cap is reached. Candidate responses are cached for efficient screening, while exact deployed-error evaluation determines acceptance. Only the selected residual blocks and the final shared factor are retained after calibration. Candidate construction, storage-aware search, and implementation details are provided in Appendix A.1.2, with complete storage accounting in Appendix A.2.

3.3 Margin-Aware Optimal Brain Routing (MOBR)

Motivation. RCBF and CERA optimize expert representations under fixed routing, making stable expert selection essential. However, conventional logit reconstruction does not directly preserve rankings determining Top-k routing. As shown in Table 1, expert-shift alone substantially degrades perplexity without expert-weight quantization. Moreover, upstream quantization perturbs router inputs, so routing preservation requires accounting for router-weight error and inherited state drift.

We propose **Margin-Aware Optimal Brain Routing (MOBR)**, which preserves selected–unselected routing margins on quantized-prefix inputs rather than minimizing logit MSE. MOBR calibrates the deployed router before RCBF and CERA, providing the fixed routing weights subsequently used by both modules. For router weights $\boldsymbol{G} \in \mathbb{R}^{d\times E}$, let $\boldsymbol{x}_t^\star$ and $\widehat{\boldsymbol{x}}_t$ denote the teacher and quantized-prefix inputs, with logits $\boldsymbol{z}_t^\star = \boldsymbol{G}^\top \boldsymbol{x}_t^\star$ and $\widehat{\boldsymbol{z}}_t = \widehat{\boldsymbol{G}}^\top \widehat{\boldsymbol{x}}_t$. For each calibration token, we construct the selected–unselected expert pairs $\mathcal{P}_t = \{(i,j) : i \in S_k(\boldsymbol{x}_t^\star),\ j \notin S_k(\boldsymbol{x}_t^\star)\}$. Their teacher and deployed margins are $m_{t,ij}^\star = z_{t,i}^\star - z_{t,j}^\star$ and $\widehat{m}_{t,ij} = \widehat{z}_{t,i} - \widehat{z}_{t,j}$, which are mapped to pairwise probabilities $q_{t,ij}^\star = \sigma(m_{t,ij}^\star/\tau)$ and $\widehat{q}_{t,ij} = \sigma(\widehat{m}_{t,ij}/\tau)$, where $\tau > 0$ is the temperature.

Table 1: Effect of expert-shift on WikiText-2 perplexity.

Model	Quantized	Expert-Shift	PPL↓
OLMoE-1B-7B	x	x	6.65
	x	✓	10.83(+4.18)
	✓	x	13.12(+6.47)
	✓	✓	15.49(+8.84)
DeepSeek-V2-Lite	x	x	6.31
	x	✓	12.21(+5.90)
	✓	x	15.33(+9.02)
	✓	✓	18.77(+12.46)

Unlike router-only reconstruction, MOBR explicitly retains both sources of logit perturbation:

$$\widehat{\boldsymbol{z}}_t - \boldsymbol{z}_t^\star = (\widehat{\boldsymbol{G}} - \boldsymbol{G})^\top \widehat{\boldsymbol{x}}_t + \boldsymbol{G}^\top(\widehat{\boldsymbol{x}}_t - \boldsymbol{x}_t^\star). \tag{11}$$

The first term captures router quantization, while the second captures inherited input drift. MOBR minimizes their effect on routing boundaries through

$$\mathcal{L}_{\mathrm{M}}(\widehat{\boldsymbol{G}}) = \frac{1}{T}\sum_t \sum_{(i,j)\in\mathcal{P}_t} D_{\mathrm{KL}}\big(\mathrm{Bern}(q_{t,ij}^\star) \,\|\, \mathrm{Bern}(\widehat{q}_{t,ij})\big)\,. \tag{12}$$

Maintaining positive selected–unselected margins preserves the teacher Top-k set, while the soft KL objective provides a differentiable surrogate for this discrete constraint. To optimize Eq. 12, we perform a local second-order expansion at the deployed router $\boldsymbol{G}^{(0)}$. Let $q_{t,ij}^{(0)}$ denote the current student pairwise probability and $\boldsymbol{a}_{ij} = \boldsymbol{e}_i - \boldsymbol{e}_j$. The per-expert gradient and block curvature are

$$\begin{aligned} \boldsymbol{g}_e &= \frac{1}{T}\sum_t b_{t,e}\widehat{\boldsymbol{x}}_t, \qquad \boldsymbol{H}_e = \frac{1}{T}\sum_t v_{t,e}\widehat{\boldsymbol{x}}_t\widehat{\boldsymbol{x}}_t^\top + \rho\boldsymbol{I}, \\ b_{t,e} &= \sum_{(i,j)\in\mathcal{P}_t} \frac{q_{t,ij}^{(0)} - q_{t,ij}^\star}{\tau}(\boldsymbol{a}_{ij})_e, \qquad v_{t,e} = \sum_{(i,j)\in\mathcal{P}_t} \frac{q_{t,ij}^{(0)}(1-q_{t,ij}^{(0)})}{\tau^2}(\boldsymbol{a}_{ij})_e^2, \end{aligned} \tag{13}$$

where $\rho > 0$ is a damping term. Unlike a teacher-centered expansion, $\boldsymbol{g}_e$ generally remains nonzero when upstream quantization shifts the router input, and the curvature is evaluated at the current student probabilities. For scalability, MOBR discards cross-column curvature blocks and optimizes

$$\sum_e \left[\boldsymbol{g}_e^\top \boldsymbol{\delta}_e + \frac{1}{2}\boldsymbol{\delta}_e^\top \boldsymbol{H}_e \boldsymbol{\delta}_e\right], \qquad \boldsymbol{\delta}_e = \widehat{\boldsymbol{G}}_{:e} - \boldsymbol{G}_{:e}^{(0)},$$

under the router's binary code-and-scale representation. Candidate updates are accepted only when the exact routing KL decreases. The resulting deployed routing weights $\bar{p}_{t,e}$ are then fixed throughout RCBF and CERA. Full coupled curvature, discrete updates, and routing-specific implementation details are provided in Appendix A.1.3.

Table 2: WikiText-2 perplexity and zero-shot accuracy on ARC-Challenge (AC), ARC-Easy (AE), HellaSwag (HS), LAMBADA-OpenAI (LO), LAMBADA-Standard (LS), PIQA (PQ), and WinoGrande (WG). #Bits(W) denotes the static target-weight BPW.

Model	Method	#Bits(W)	AE↑	AC↑	HS↑	LO↑	LS↑	PQ↑	WG↑	Avg.↑	Wiki PPL↓
OLMoE-1B-7B	Baseline	16	76.77	49.23	78.21	70.83	65.57	79.76	68.82	69.88	6.65
	AWQ	3	31.18	27.32	31.22	4.21	0.55	51.02	50.33	27.98	7e2
	GPTQ	3	59.78	31.22	49.61	45.12	25.88	69.29	58.54	48.49	14.99
	AWQ	2	25.05	24.11	24.43	0	0	50.31	50.88	24.97	5e4
	GPTQ	2	25.45	24.45	26.93	0	0	52.08	51.43	25.76	2e3
	MoEQuant	2	35.54	26.09	36.55	13.22	10.11	58.77	51.22	33.07	6e2
	NoWag	2.08	60.16	**34.64**	51.35	30.90	21.77	65.82	55.09	45.68	15.02
	BiLLM	1.11	44.19	24.91	37.40	18.53	14.13	59.58	51.46	35.74	20.32
	ARB-LLM	1.11	46.51	26.96	43.45	32.31	24.98	63.33	57.62	42.16	15.49
	BitMoE	1.11	**62.85**	34.46	**53.24**	**48.61**	**30.75**	**69.60**	**60.35**	**51.41**	**13.76**
DeepSeek-V2-Lite	Baseline	16	74.03	46.16	73.55	68.37	61.36	79.27	67.64	67.20	6.31
	AWQ	3	28.18	26.38	31.12	2.12	0.26	50.99	51.27	27.19	2e3
	GPTQ	3	45.33	25.22	42.64	33.22	25.88	65.11	53.41	41.54	16.26
	AWQ	2	25.01	25.11	25.41	0	0	50.21	50.52	25.18	6e7
	GPTQ	2	25.58	**27.02**	27.47	0	0	51.87	51.51	26.21	5e7
	MoEQuant	2	32.11	24.22	36.05	10.21	8.20	55.27	51.92	31.14	9e2
	NoWag	2.10	45.51	25.53	40.88	21.21	19.31	59.86	54.96	38.18	17.22
	BiLLM	1.11	39.44	23.81	36.39	16.61	19.10	60.12	52.96	35.49	22.43
	ARB-LLM	1.11	44.82	26.28	38.97	26.74	27.75	61.59	54.06	40.03	18.77
	BitMoE	1.11	**50.09**	27.01	**44.31**	**37.27**	**29.67**	**67.04**	**57.03**	**44.63**	**15.31**
Qwen3-30B-A3B	Baseline	16	78.96	55.97	77.64	64.86	63.26	80.47	69.53	70.10	8.70
	AWQ	3	38.17	27.32	37.12	4.12	1.22	51.32	51.47	30.11	7e2
	GPTQ	3	65.32	43.29	56.62	48.22	41.70	70.19	65.48	55.83	14.21
	AWQ	2	25.08	22.70	25.04	0	0	49.51	49.57	24.56	4e4
	GPTQ	2	24.24	26.88	25.95	0	0	50.27	50.20	25.36	2e4
	MoEQuant	2	32.33	24.76	34.41	12.12	10.15	53.77	51.99	31.36	4e2
	NoWag	2.11	65.50	35.53	50.31	42.99	39.31	69.86	64.96	52.64	15.03
	BiLLM	1.11	32.45	23.55	35.35	17.19	14.56	55.17	52.80	33.01	52.78
	ARB-LLM	1.11	44.15	27.30	44.63	24.72	20.05	63.93	56.20	40.14	26.76
	BitMoE	1.11	**68.43**	**45.15**	**61.62**	**54.49**	**47.59**	**73.96**	**67.72**	**59.85**	**12.26**

4 EXPERIMENTS

4.1 EXPERIMENT SETUP

Models and Datasets. We evaluate BitMoE across six open-source MoE-based LLMs spanning diverse architectures and scales: **OLMoE** (Muennighoff et al., 2025), **DeepSeek-V2-Lite** (DeepSeek-AI et al., 2024), **Qwen3-30B-A3B** (Yang et al., 2025), **Qwen3-Next-80B-A3B-Instruct** (Yang et al., 2025), **Qwen3.6-35B-A3B** (Qwen Team, 2026), and **Gemma-4-26B-A4B-it** (Google DeepMind, 2026). To comprehensively assess performance, we measure perplexity on WikiText2 (Merity et al., 2016) and zero-shot accuracy across seven datasets: ARC-Challenge, ARC-Easy (Clark et al., 2018), HellaSwag (Zellers et al., 2019), LAMBADA-OpenAI, LAMBADA-Standard (Paperno et al., 2016), PIQA (Bisk et al., 2019), and WinoGrande (Sakaguchi et al., 2019). Furthermore, we evaluate BitMoE on more challenging reasoning benchmarks, including the multi-domain knowledge task MMLU (Hendrycks et al., 2021), the mathematical reasoning dataset GSM8K (Cobbe et al., 2021), and the code generation task HumanEval (Chen et al., 2021).

Baseline Methods. We compare BitMoE with representative baselines for binary large language models, including BiLLM (Huang et al., 2024) and ARB-LLM (Li et al., 2024). We also evaluate widely adopted PTQ methods, namely AWQ (Lin et al., 2024) and GPTQ (Frantar et al., 2023), as well as MoEQuant (Hu et al., 2025), which is specifically designed for MoE-based LLMs. In addition, we include the vector quantization method NoWag (Liu et al., 2025), which targets extremely low-bit regimes. Higher-bit quantization methods (e.g., 4-bit and 8-bit) are excluded, as they are not directly comparable to binarization.

Implementation Details. All experiments are conducted on NVIDIA RTX PRO 6000 GPUs with a sequence length of 4,096, using 128 randomly sampled WikiText2 sequences for calibration. We retain the shared factors in 8-bit precision and use ARB-LLM-style alternating reconstruction as the binary fitting operator. BitMoE performs layerwise offline calibration: MOBR first determines deployed routing, RCBF refines the shared and binary expert factors, and CERA allocates binary residuals under storage budget. No full-model fine-tuning is required. Unless otherwise stated, the hyperparameters used in our experiments are summarized in Appendix A.4.1. We use lm-eval (Gao et al., 2024) for the seven zero-shot tasks and EvalScope (Team, 2024a) for MMLU, GSM8K, and HumanEval. Reported effective bitwidth includes deployed binary payloads, shared factors, scales, residuals, and metadata within the target-weight scope; see Appendix A.2 for detailed accounting.

4.2 Main Results

Comparison Results. We systematically evaluate extreme low-bit quantization across MoE-based LLMs and benchmarks. As shown in Table 2, BitMoE achieves the best average zero-shot accuracy and lowest perplexity among compressed methods across three models, while leading most individual tasks. Conventional quantizers AWQ and GPTQ degrade severely at 2-bit precision, whereas dense-LLM binarization methods BiLLM and ARB-LLM exhibit substantial performance gaps on MoE architectures. In contrast, BitMoE preserves language modeling and zero-shot capabilities. Notably, at the same 1.11 static BPW as BiLLM and ARB-LLM, BitMoE improves the accuracy–compression trade-off. On Qwen3-30B-A3B, it reduces perplexity by 54.2% (26.76 $\rightarrow$ 12.26) and improves average zero-shot accuracy by 19.71 percentage points (40.14% $\rightarrow$ 59.85%) over ARB-LLM. It also surpasses 3-bit GPTQ in both perplexity and average accuracy despite a substantially lower static bitwidth. Model-specific active-path and whole-model BPW analyses, together with detailed dialogue examples, are provided in Appendix A.4.2 and Appendix A.5, respectively.

Experiments on Instruction-tuned Models. Instruction tuning improves LLM utility but complicates extreme quantization, as delicate alignment distributions are highly sensitive to weight perturbations. We evaluate BitMoE on Qwen3-Next-80B-A3B-Instruct and Gemma-4-26B-A4B-it across three reasoning benchmarks (Table 3). Under extreme low-bit quantization, 2-bit GPTQ collapses severely, with MMLU approaching random-guess performance and both HumanEval and GSM8K dropping to zero. Binary baselines such as BiLLM and ARB-LLM also exhibit substantial degradation. In contrast, BitMoE at only 1.08–1.10 static BPW effectively preserves the reasoning capability of the original models. It consistently outperforms the binary baselines and surpasses 3-bit GPTQ across all three benchmarks on both models, while using less than half its static target-weight BPW. These results demonstrate the robustness of BitMoE on instruction-tuned MoEs.

Table 3: Results on instruction-tuned models across MMLU, HumanEval, and GSM8K.

Model	Method	#Bits(W)	MMLU	HumanEval	GSM8K
Qwen3-Next-80B-A3B-Instruct	BF16	16	88.77	94.51	96.44
	GPTQ	3	69.15	54.01	59.27
	GPTQ	2	24.83	0	0
	BiLLM	1.06	52.04	25.20	33.81
	ARB-LLM	1.06	64.90	38.59	48.11
	BitMoE	**1.08**	**74.62**	**60.67**	**64.71**
Gemma-4-26B-A4B-it	BF16	16	89.36	95.12	96.97
	GPTQ	3	69.88	52.47	58.20
	GPTQ	2	26.52	0	0
	BiLLM	1.09	51.84	22.66	31.40
	ARB-LLM	1.09	65.76	38.27	46.33
	BitMoE	**1.10**	**75.19**	**57.98**	**64.66**

4.3 Ablation Studies

BitMoE integrates RCBF, CERA, and MOBR to refine binary representations, allocate residual capacity, and preserve routing. Our ablations examine (i) the individual and combined effects of the three components and (ii) MOBR's effectiveness in suppressing expert-shift through routing consistency. Additional studies of calibration settings, MOBR's temperature τ, and shared-factor precision are provided in Appendix A.4.7, Appendix A.4.8, and Appendix A.4.4, respectively.

Modular Sensitivity Study. Table 4 compares different component combinations on Qwen3.6-35B-A3B. All variants retain joint-SVD initialization, ranks, stored precisions, and layerwise storage caps; disabling RCBF removes joint refinement rather than the shared parameterization. Without CERA, residuals follow a fixed magnitude ranking with shared-factor readaptation; without MOBR, router calibration uses logit MSE. RCBF provides the largest individual accuracy gain, while the full configuration achieves the best overall performance, reaching 11.23 WikiText-2 perplexity, 63.13%

average zero-shot accuracy, and 65.48% MMLU. Compared with the control, the latter two improve by 32.92 and 36.22 percentage points, respectively. Further analyses of RCBF decomposition and CERA residual allocation are provided in Appendix A.4.3 and Appendix A.4.5, respectively.

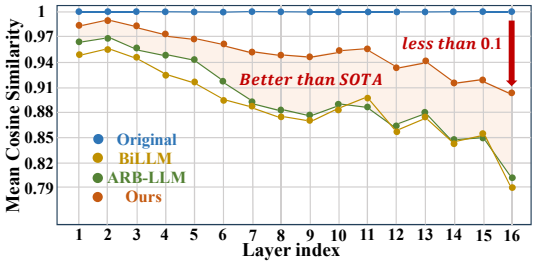


Figure 5: Cosine similarity between BF16 and binary-model routing probabilities.

Table 4: Impact of different components in BitMoE on Qwen3.6-35B-A3B.

Modules			Qwen3.6-35B-A3B		
RCBF	CERA	MOBR	Wiki PPL↓	Avg.7↑	MMLU↑
–	–	–	100.00	30.21	29.26
✓	–	–	19.53	49.39	48.97
–	✓	–	32.12	44.19	42.85
–	–	✓	26.38	43.22	38.39
✓	✓	–	14.40	57.73	58.25
✓	✓	✓	**11.23**	**63.13**	**65.48**

Routing Consistency Analysis. We further evaluate MOBR through routing consistency S_ℓ, defined as the mean cosine similarity between quantized and full-precision routing probabilities. As shown in Fig. 5, BiLLM and ARB-LLM exhibit substantial routing degradation in deeper layers, whereas BitMoE with MOBR maintains $S_\ell \geq 0.9$ across layers, indicating substantially closer agreement with full-precision routing. Additional routing analyses and the integration of MOBR into baseline quantizers are provided in Appendix A.4.6 and Appendix A.4.9, respectively.

4.4 Efficiency Analysis of BitMoE

Speedup and Memory Savings. We benchmarked BitMoE's system efficiency on NVIDIA RTX PRO 6000 GPUs, comparing throughput and per-token active weight memory between Qwen3-30B-A3B and its binarized counterpart under different shared-factor precisions. With a batch size of 4, prefill length of 2,048 tokens, and decoding length of 1,024 tokens, Fig. 6 (left) shows that under the default 8-bit shared-factor setting, BitMoE achieves a static target-weight BPW of 1.11 and an active-path BPW of 1.72. Its per-token active weight memory decreases by 72.7%, from 6.71 GB in BF16 to 1.83 GB. Meanwhile, decoding throughput increases from 42.17 to 98.52 tokens/s, corresponding to a 2.34× speedup. These results demonstrate substantial efficiency gains under extreme weight compression.

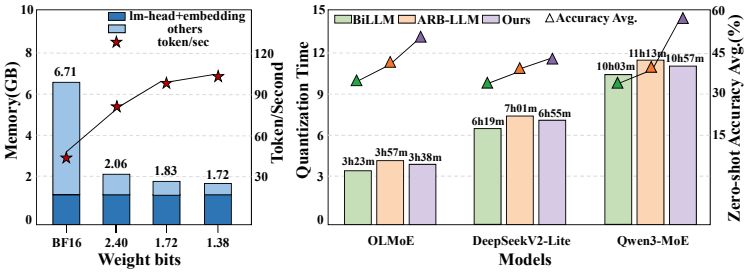


Figure 6: **Left**: Throughput vs. active weight memory for BF16 and BitMoE across varying bitwidths. **Right**: Trade-off between quantization time and zero-shot accuracy among binarization baselines.

Quantization Time Comparison. Fig. 6 (right) compares quantization cost and zero-shot accuracy across binarization methods. Although RCBF and CERA introduce iterative refinement and conditional residual allocation, BitMoE maintains practical calibration efficiency, requiring 3h38m, 6h55m, and 10h57m on OLMoE, DeepSeekV2-Lite, and Qwen3-MoE, respectively. It remains faster than ARB-LLM across all three models, while achieving substantially higher average zero-shot accuracy than both BiLLM and ARB-LLM. These results demonstrate that BitMoE yields a favorable trade-off between calibration cost and task performance.

5 Conclusion

We present BitMoE, a post-training binarization framework tailored to MoE-based LLMs. By jointly improving binary expert representation, allocating residual capacity conditionally, and preserving routing decisions, BitMoE achieves a strong accuracy–efficiency trade-off. Experiments show consistent gains over prior binary PTQ methods together with substantial storage reduction and inference speedup, supporting practical MoE deployment.

AI USE STATEMENT

Generative AI tools were used during manuscript preparation only for language-related assistance, including proofreading, translation, concision, and improvements to readability. They were not used to generate the core research ideas, methodological contributions, experimental protocols, or scientific conclusions. Any AI-supported edits were manually inspected and, when necessary, revised by the authors before inclusion in the manuscript. Responsibility for the technical content, reported results, and final presentation rests entirely with the authors.

ETHICS STATEMENT

This work studies BitMoE, a post-training binarization framework for reducing the memory and inference costs of Mixture-of-Experts large language models. Our experiments use existing pretrained models and public calibration and evaluation benchmarks. Lowering deployment cost can improve accessibility and energy efficiency, but compressed models may retain biases, factual errors, unsafe behaviors, and other limitations of their full-precision counterparts, while quantization can introduce additional behavioral changes not captured by standard accuracy benchmarks. Our experiments evaluate model quality and system efficiency rather than safety, fairness, privacy, or robustness guarantees. Deployment should therefore include application-specific evaluation, appropriate safeguards, and compliance with the licenses and usage policies of the underlying models and datasets.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, Sec. 3 describes BitMoE and its three core components, while Sec. 4 specifies the models, datasets, calibration procedure, baselines, evaluation protocol, and hardware. Appendix Secs. A.1 and A.3 provide detailed derivations and pseudocode, while Appendix Secs. A.4.1 and A.4 report implementation hyperparameters and additional analyses. Together, these details specify the complete post-training binarization and evaluation pipeline. We plan to release the implementation code, configurations, and evaluation scripts after the review process.

REFERENCES

Allen Institute for AI. OLMoE-1B-7B-0125: Official checkpoint configuration. Hugging Face model repository, 2025. URL https://huggingface.co/allenai/OLMoE-1B-7B-0125/blob/main/config.json. Accessed September 26, 2026.

Haoli Bai, Wei Zhang, Lu Hou, Lifeng Shang, Jing Jin, Xin Jiang, Qun Liu, Michael Lyu, and Irwin King. Binarybert: Pushing the limit of bert quantization, 2021. URL https://arxiv.org/abs/2012.15701.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language, 2019. URL https://arxiv.org/abs/1911.11641.

Vladimír Boža and Vladimír Macko. Addition is almost all you need: Compressing neural networks with double binary factorization, 2025. URL https://arxiv.org/abs/2505.11076.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

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, 2018. URL https://arxiv.org/abs/1803.05457.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Damai Dai, Chengqi Deng, Chenggang Zhao, R. X. Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y. Wu, Zhenda Xie, Y. K. Li, Panpan Huang, Fuli Luo, Chong Ruan, Zhifang Sui, and Wenfeng Liang. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models, 2024. URL https://arxiv.org/abs/2401.06066.

DeepSeek-AI. DeepSeek-V2-Lite: Official checkpoint configuration. Hugging Face model repository, 2024a. URL https://huggingface.co/deepseek-ai/DeepSeek-V2-Lite/blob/main/config.json. Accessed September 26, 2026.

DeepSeek-AI. DeepSeek-V2-Lite: Official model implementation. Hugging Face model repository, 2024b. URL https://huggingface.co/deepseek-ai/DeepSeek-V2-Lite/blob/main/modeling_deepseek.py. Accessed September 26, 2026.

DeepSeek-AI, Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Dengr, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Hao Yang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jin Chen, Jingyang Yuan, Junjie Qiu, Junxiao Song, Kai Dong, Kaige Gao, Kang Guan, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruizhe Pan, Runxin Xu, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Size Zheng, T. Wang, Tian Pei, Tian Yuan, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Liu, Xin Xie, Xingkai Yu, Xinnan Song, Xinyi Zhou, Xinyu Yang, Xuan Lu, Xuecheng Su, Y. Wu, Y. K. Li, Y. X. Wei, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Zheng, Yichao Zhang, Yiliang Xiong, Yilong Zhao, Ying He, Ying Tang, Yishi Piao, Yixin Dong, Yixuan Tan, Yiyuan Liu, Yongji Wang, Yongqiang Guo, Yuchen Zhu, Yuduan Wang, Yuheng Zou, Yukun Zha, Yunxian Ma, Yuting Yan, Yuxiang You, Yuxuan Liu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhewen Hao, Zhihong Shao, Zhiniu Wen, Zhipeng Xu, Zhongyu Zhang, Zhuoshu Li, Zihan Wang, Zihui Gu, Zilin Li, and Ziwei Xie. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model, 2024. URL https://arxiv.org/abs/2405.04434.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao,

Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. Deepseek-v3 technical report, 2025. URL https://arxiv.org/abs/2412.19437.

Peijie Dong, Lujun Li, Yuedong Zhong, Dayou Du, Ruibo Fan, Yuhan Chen, Zhenheng Tang, Qiang Wang, Wei Xue, Yike Guo, and Xiaowen Chu. Stbllm: Breaking the 1-bit barrier with structured binary llms, 2024. URL https://arxiv.org/abs/2408.01803.

Haojie Duanmu, Xiuhong Li, Zhihang Yuan, Size Zheng, Jiangfei Duan, Xingcheng Zhang, and Dahua Lin. Mxmoe: Mixed-precision quantization for moe with accuracy and performance co-design, 2025. URL https://arxiv.org/abs/2505.05799.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity, 2022. URL https://arxiv.org/abs/2101.03961.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers, 2023. URL https://arxiv.org/abs/2210.17323.

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. The language model evaluation harness, 07 2024. URL https://zenodo.org/records/12608602.

Google DeepMind. Gemma 4: Our most capable open models to date. https://blog.google/innovation-and-ai/technology/developers-tools/gemma-4/, April 2026. Accessed: 2026-04-19.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021. URL https://arxiv.org/abs/2009.03300.

Xing Hu, Zhixuan Chen, Dawei Yang, Zukang Xu, Chen Xu, Zhihang Yuan, Sifan Zhou, and Jiangyong Yu. Moequant: Enhancing quantization for mixture-of-experts large language models via expert-balanced sampling and affinity guidance, 2025. URL https://arxiv.org/abs/2505.03804.

Wei Huang, Yangdong Liu, Haotong Qin, Ying Li, Shiming Zhang, Xianglong Liu, Michele Magno, and Xiaojuan Qi. Billm: Pushing the limit of post-training quantization for llms, 2024. URL https://arxiv.org/abs/2402.04291.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991. doi: 10.1162/neco.1991.3.1.79.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le

Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mixtral of experts, 2024. URL https://arxiv.org/abs/2401.04088.

Michael I. Jordan and Robert A. Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural Computation*, 6(2):181–214, 1994. doi: 10.1162/neco.1994.6.2.181.

Jinuk Kim, Marwa El Halabi, Wonpyo Park, Clemens JS Schaefer, Deokjae Lee, Yeonhong Park, Jae W. Lee, and Hyun Oh Song. Guidedquant: Large language model quantization via exploiting end loss guidance, 2025. URL https://arxiv.org/abs/2505.07004.

Lujun Li, Zhu Qiyuan, Jiacheng Wang, Wei Li, Hao Gu, Sirui Han, and Yike Guo. Sub-moe: Efficient mixture-of-expert llms compression via subspace expert merging, 2025. URL https://arxiv.org/abs/2506.23266.

Zhiteng Li, Xianglong Yan, Tianao Zhang, Haotong Qin, Dong Xie, Jiang Tian, zhongchao shi, Linghe Kong, Yulun Zhang, and Xiaokang Yang. Arb-llm: Alternating refined binarizations for large language models, 2024. URL https://arxiv.org/abs/2410.03129.

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 llm compression and acceleration, 2024. URL https://arxiv.org/abs/2306.00978.

Lawrence Liu, Inesh Chakrabarti, Yixiao Li, Mengdi Wang, Tuo Zhao, and Lin F. Yang. Nowag: A unified framework for shape preserving compression of large language models, 2025. URL https://arxiv.org/abs/2504.14569.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

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, 2025. URL https://arxiv.org/abs/2409.02060.

OpenAI, :, Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K. Arora, Yu Bai, Bowen Baker, Haiming Bao, Boaz Barak, Ally Bennett, Tyler Bertao, Nivedita Brett, Eugene Brevdo, Greg Brockman, Sebastien Bubeck, Che Chang, Kai Chen, Mark Chen, Enoch Cheung, Aidan Clark, Dan Cook, Marat Dukhan, Casey Dvorak, Kevin Fives, Vlad Fomenko, Timur Garipov, Kristian Georgiev, Mia Glaese, Tarun Gogineni, Adam Goucher, Lukas Gross, Katia Gil Guzman, John Hallman, Jackie Hehir, Johannes Heidecke, Alec Helyar, Haitang Hu, Romain Huet, Jacob Huh, Saachi Jain, Zach Johnson, Chris Koch, Irina Kofman, Dominik Kundel, Jason Kwon, Volodymyr Kyrylov, Elaine Ya Le, Guillaume Leclerc, James Park Lennon, Scott Lessans, Mario Lezcano-Casado, Yuanzhi Li, Zhuohan Li, Ji Lin, Jordan Liss, Lily, Liu, Jiancheng Liu, Kevin Lu, Chris Lu, Zoran Martinovic, Lindsay McCallum, Josh McGrath, Scott McKinney, Aidan McLaughlin, Song Mei, Steve Mostovoy, Tong Mu, Gideon Myles, Alexander Neitz, Alex Nichol, Jakub Pachocki, Alex Paino, Dana Palmie, Ashley Pantuliano, Giambattista Parascandolo, Jongsoo Park, Leher Pathak, Carolina Paz, Ludovic Peran, Dmitry Pimenov, Michelle Pokrass, Elizabeth Proehl, Huida Qiu, Gaby Raila, Filippo Raso, Hongyu Ren, Kimmy Richardson, David Robinson, Bob Rotsted, Hadi Salman, Suvansh Sanjeev, Max Schwarzer, D. Sculley, Harshit Sikchi, Kendal Simon, Karan Singhal, Yang Song, Dane Stuckey, Zhiqing Sun, Philippe Tillet, Sam Toizer, Foivos Tsimpourlas, Nikhil Vyas, Eric Wallace, Xin Wang, Miles Wang, Olivia Watkins, Kevin Weil, Amy Wendling, Kevin Whinnery, Cedric Whitney, Hannah Wong, Lin Yang, Yu Yang, Michihiro Yasunaga, Kristen Ying, Wojciech Zaremba, Wenting Zhan, Cyril Zhang, Brian Zhang, Eddie Zhang, and Shengjia Zhao. gpt-oss-120b & gpt-oss-20b model card, 2025. URL https://arxiv.org/abs/2508.10925.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The lambada dataset: Word prediction requiring a broad discourse context, 2016. URL https://arxiv.org/abs/1606.06031.

Qwen Team. Qwen3-30B-A3B: Official checkpoint configuration. Hugging Face model repository, 2025. URL https://huggingface.co/Qwen/Qwen3-30B-A3B/blob/main/config.json. Accessed September 26, 2026.

Qwen Team. Qwen3.6-35B-A3B: Agentic coding power, now open to all, April 2026. URL https://qwen.ai/blog?id=qwen3.6-35b-a3b.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale, 2019. URL https://arxiv.org/abs/1907.10641.

Yuzhang Shang, Zhihang Yuan, Qiang Wu, and Zhen Dong. Pb-llm: Partially binarized large language models, 2023. URL https://arxiv.org/abs/2310.00034.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, Zhuofu Chen, Jialei Cui, Hao Ding, Mengnan Dong, Angang Du, Chenzhuang Du, Dikang Du, Yulun Du, Yu Fan, Yichen Feng, Kelin Fu, Bofei Gao, Hongcheng Gao, Peizhong Gao, Tong Gao, Xinran Gu, Longyu Guan, Haiqing Guo, Jianhang Guo, Hao Hu, Xiaoru Hao, Tianhong He, Weiran He, Wenyang He, Chao Hong, Yangyang Hu, Zhenxing Hu, Weixiao Huang, Zhiqi Huang, Zihao Huang, Tao Jiang, Zhejun Jiang, Xinyi Jin, Yongsheng Kang, Guokun Lai, Cheng Li, Fang Li, Haoyang Li, Ming Li, Wentao Li, Yanhao Li, Yiwei Li, Zhaowei Li, Zheming Li, Hongzhan Lin, Xiaohan Lin, Zongyu Lin, Chengyin Liu, Chenyu Liu, Hongzhang Liu, Jingyuan Liu, Junqi Liu, Liang Liu, Shaowei Liu, T. Y. Liu, Tianwei Liu, Weizhou Liu, Yangyang Liu, Yibo Liu, Yiping Liu, Yue Liu, Zhengying Liu, Enzhe Lu, Lijun Lu, Shengling Ma, Xinyu Ma, Yingwei Ma, Shaoguang Mao, Jie Mei, Xin Men, Yibo Miao, Siyuan Pan, Yebo Peng, Ruoyu Qin, Bowen Qu, Zeyu Shang, Lidong Shi, Shengyuan Shi, Feifan Song, Jianlin Su, Zhengyuan Su, Xinjie Sun, Flood Sung, Heyi Tang, Jiawen Tao, Qifeng Teng, Chensi Wang, Dinglu Wang, Feng Wang, Haiming Wang, Jianzhou Wang, Jiaxing Wang, Jinhong Wang, Shengjie Wang, Shuyi Wang, Yao Wang, Yejie Wang, Yiqin Wang, Yuxin Wang, Yuzhi Wang, Zhaoji Wang, Zhengtao Wang, Zhexu Wang, Chu Wei, Qianqian Wei, Wenhao Wu, Xingzhe Wu, Yuxin Wu, Chenjun Xiao, Xiaotong Xie, Weimin Xiong, Boyu Xu, Jing Xu, Jinjing Xu, L. H. Xu, Lin Xu, Suting Xu, Weixin Xu, Xinran Xu, Yangchuan Xu, Ziyao Xu, Junjie Yan, Yuzi Yan, Xiaofei Yang, Ying Yang, Zhen Yang, Zhilin Yang, Zonghan Yang, Haotian Yao, Xingcheng Yao, Wenjie Ye, Zhuorui Ye, Bohong Yin, Longhui Yu, Enming Yuan, Hongbang Yuan, Mengjie Yuan, Haobing Zhan, Dehao Zhang, Hao Zhang, Wanlu Zhang, Xiaobin Zhang, Yangkun Zhang, Yizhi Zhang, Yongting Zhang, Yu Zhang, Yutao Zhang, Yutong Zhang, Zheng Zhang, Haotian Zhao, Yikai Zhao, Huabin Zheng, Shaojie Zheng, Jianren Zhou, Xinyu Zhou, Zaida Zhou, Zhen Zhu, Weiyu Zhuang, and Xinxing Zu. Kimi k2: Open agentic intelligence, 2025. URL https://arxiv.org/abs/2507.20534.

ModelScope Team. EvalScope: Evaluation framework for large models, 2024a. URL https://github.com/modelscope/evalscope.

Qwen Team. Qwen1.5-moe-a2.7b: A sparse mixture-of-experts pre-trained language model. https://huggingface.co/Qwen/Qwen1.5-MoE-A2.7B, 2024b. Pre-trained language model; Accessed: [n. d.].

Zukang Xu, Zhixiong Zhao, Xing Hu, Zhixuan Chen, and Dawei Yang. Kbvq-moe: Klt-guided svd with bias-corrected vector quantization for moe large language models. *arXiv preprint arXiv:2602.11184*, 2026.

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, 2025. URL https://arxiv.org/abs/2505.09388.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence?, 2019. URL https://arxiv.org/abs/1905.07830.

APPENDIX OVERVIEW

- Section A.1: Detailed Derivations and Implementation Details.
  - Section A.1.1: Routing-Coupled Binary Factorization.
  - Section A.1.2: Conditional Expert Residual Allocation.
  - Section A.1.3: Input-Aware Margin Preservation in MOBR.
- Section A.2: Deployment Storage and Effective Bitwidth.
  - Section A.2.1: Fixed Representation and Resident Storage.
  - Section A.2.2: Residual Payloads and Incremental Costs.
  - Section A.2.3: Active Payload, Resident Weights, and Runtime Memory.
  - Section A.2.4: Whole-Model Bitwidth and Comparison Scope.
- Section A.3: Layerwise Calibration and Pseudocode.
  - Section A.3.1: Overall Layerwise Workflow.
  - Section A.3.2: MOBR Router Calibration.
  - Section A.3.3: RCBF Base-Representation Refinement.
  - Section A.3.4: Construction of Fixed Residual Candidates.
  - Section A.3.5: Conditional Residual Selection and Readaptation.
  - Section A.3.6: Binary Fitting Primitive and Retained Scales.
  - Section A.3.7: Calibration Configuration and Numerical Safeguards.
- Section A.4: More Experimental Results.
  - Section A.4.1: Calibration and Search Hyperparameters.
  - Section A.4.2: Model-Specific Active-Path and Whole-Model Bitwidth.
  - Section A.4.3: Comparison with Joint-SVD-Based Expert Decomposition.
  - Section A.4.4: Shared-Factor Precision and Bitwidth Analysis.
  - Section A.4.5: Comparison of Residual Allocation Strategies.
  - Section A.4.6: Ablation Study on Routing Similarity.
  - Section A.4.7: Ablation Study on Calibration Data.
  - Section A.4.8: Ablation Study on the Temperature Parameter τ.
  - Section A.4.9: Plug-and-Play Ablations of MOBR.
- Section A.5: Dialog Examples.
- Section A.6: Limitations.
- Section A.7: Broader Impacts.

A APPENDIX

A.1 DETAILED DERIVATIONS AND IMPLEMENTATION DETAILS

This section expands the RCBF, CERA, and MOBR formulations in Sec. 3. We derive the deployed objectives, storage costs, and layerwise calibration updates, distinguishing exact finite-loss evaluations from local optimization surrogates. Model-specific ranks and storage formats are given in Appendix A.4.2, and the corresponding calibration algorithms are presented in Appendix A.3.

A.1.1 ROUTING-COUPLED BINARY FACTORIZATION

Projection-wise initialization and dimensions. For one MoE layer, suppress the layer index and use E for the number of routed experts. For each projection type $m \in \{\mathrm{gate}, \mathrm{up}, \mathrm{down}\}$, concatenate matrices with the same shape:

$$\overline{\boldsymbol{W}}_m = [\boldsymbol{W}_1^{(m)}, \ldots, \boldsymbol{W}_E^{(m)}] \in \mathbb{R}^{O_m \times EI_m}, \qquad \overline{\boldsymbol{W}}_m = \boldsymbol{U}_m \boldsymbol{\Sigma}_m \boldsymbol{P}_m^\top. \tag{14}$$

For a fixed retained rank $1 \le r_m \le \min(O_m, EI_m)$, initialize

$$\begin{aligned} \boldsymbol{A}_m^{\mathrm{svd}} &= \boldsymbol{U}_m[:, 1:r_m]\boldsymbol{\Sigma}_m[1:r_m, 1:r_m], \\ [\boldsymbol{V}_1^{(m,0)}, \ldots, \boldsymbol{V}_E^{(m,0)}] &= \boldsymbol{P}_m^\top[1:r_m, :]. \end{aligned} \tag{15}$$

Each $\boldsymbol{V}_e^{(m,0)}$ has shape $r_m \times I_m$ and $\boldsymbol{A}_m$ has shape $O_m \times r_m$. For column inputs, the forward map is $\boldsymbol{A}_m(\boldsymbol{V}_e^{(m)}\boldsymbol{u})$; for a row-batched input $\boldsymbol{X}$, it is $\boldsymbol{X}(\boldsymbol{V}_e^{(m)})^\top \boldsymbol{A}_m^\top$. No transpose is applied to the entire horizontal block list in place of the individual factors. The maximum admissible retained rank is not necessarily the algebraic rank of $\overline{\boldsymbol{W}}_m$. Ranks are fixed before calibration and are not reallocated by CERA.

What SVD does and does not optimize. Truncated SVD minimizes the unweighted rank-constrained weight reconstruction error, whose value is $\sum_{j>r_m} \sigma_{m,j}^2$. It is not in general the minimizer of activation-weighted reconstruction or routed nonlinear output error. For an input second-moment matrix $\boldsymbol{C}_e = \mathbb{E}[\boldsymbol{u}_e\boldsymbol{u}_e^\top]$ and weight error $\Delta\boldsymbol{W}_e$, the activation error is

$$\mathbb{E}\|\Delta\boldsymbol{W}_e\boldsymbol{u}_e\|_2^2 = \operatorname{tr}(\Delta\boldsymbol{W}_e\boldsymbol{C}_e\Delta\boldsymbol{W}_e^\top). \tag{16}$$

A common isotropic second moment reduces the summed activation error to a common multiple of weight-space error; this condition is not assumed. Moreover, if $\boldsymbol{A} = \boldsymbol{U}_r\boldsymbol{\Sigma}_r$ and $\Delta\boldsymbol{V}$ is the binary-factor error, then

$$\mathbb{E}\|\boldsymbol{A}\Delta\boldsymbol{V}\boldsymbol{u}\|_2^2 = \operatorname{tr}(\boldsymbol{\Sigma}_r^2\Delta\boldsymbol{V}\boldsymbol{C}\Delta\boldsymbol{V}^\top). \tag{17}$$

Thus, an orthogonal left factor alone does not guarantee small binary output error. RCBF uses this decomposition as an initial coordinate system and subsequently optimizes the deployed representation.

Binary factors and deployment rounding. The first-plane factor is

$$\widehat{\boldsymbol{V}}_e^{(m,1)} = \operatorname{diag}(\boldsymbol{s}_{e,m}^r)\boldsymbol{B}_{e,m}\operatorname{diag}(\boldsymbol{s}_{e,m}^c), \qquad \boldsymbol{B}_{e,m} \in \{-1,+1\}^{r_m \times I_m}. \tag{18}$$

Let $\mathcal{Q}_{h,m}$ denote quantization followed by dequantization in the prescribed shared-factor format, and $\mathcal{Q}_s$ the corresponding rounding of retained scales. Symbols such as $\boldsymbol{A}_m$ and $\boldsymbol{s}_{e,m}^r$ denote the numerical values recovered from these stored representations, not permanent full-precision copies. A row/column binary fit initializes each factor using the primitive in Appendix A.3.6. During RCBF, signs, scales, and shared factors are refined using the output objective instead of continuing to optimize only the initialization loss. Scale format, quantizer metadata, and packing rules remain fixed.

Complete expert functions and paired calibration states. For a standard gated expert, a representative computation is

$$\begin{aligned} \boldsymbol{h}_{t,e} &= \phi\Big(\boldsymbol{A}_{\text{gate}}\widehat{\boldsymbol{V}}_e^{(\text{gate})}\widehat{\boldsymbol{x}}_t + \boldsymbol{b}_e^g\Big) \odot \Big(\boldsymbol{A}_{\text{up}}\widehat{\boldsymbol{V}}_e^{(\text{up})}\widehat{\boldsymbol{x}}_t + \boldsymbol{b}_e^u\Big), \\ \widehat{f}_e(\widehat{\boldsymbol{x}}_t) &= \boldsymbol{A}_{\text{down}}\widehat{\boldsymbol{V}}_e^{(\text{down})}\boldsymbol{h}_{t,e} + \boldsymbol{b}_e^d. \end{aligned} \tag{19}$$

Original biases are retained when present and are zero otherwise; the actual checkpoint's activation function and any additional fixed operations are preserved. Equation (19) is not a replacement for a different expert architecture. Teacher targets $\boldsymbol{y}_t^\star$ use teacher states $\boldsymbol{x}_t^\star$ on the same token sequences, whereas student experts use fixed quantized-prefix states $\widehat{\boldsymbol{x}}_t$. Let $\boldsymbol{c}_t$ contain any fixed always-active branch outputs evaluated on student states. Here T denotes the number of calibration tokens used for the current layer. After MOBR, the actual model router supplies $\bar{p}_{t,e}$, including its original normalization and selection rules. Pairwise sigmoid probabilities used by MOBR are not substituted for these aggregation weights. The RCBF loss is

$$\mathcal{L}_{\text{R}}(\Theta) = \frac{1}{T}\sum_t \left\| \boldsymbol{y}_t^\star - \boldsymbol{c}_t - \sum_e \bar{p}_{t,e}\widehat{f}_e(\widehat{\boldsymbol{x}}_t;\Theta) \right\|_2^2. \tag{20}$$

All teacher targets, prefix states, fixed branches, and routing weights remain unchanged within this layer's RCBF and CERA optimization.

Expert interaction and conditional targets. For arbitrary expert-output perturbations $\boldsymbol{\delta}_{t,e}$ under fixed routing,

$$\begin{aligned} \left\| \sum_e \bar{p}_{t,e}\boldsymbol{\delta}_{t,e} \right\|_2^2 = \; & \sum_e \bar{p}_{t,e}^2\|\boldsymbol{\delta}_{t,e}\|_2^2 \\ & + 2\sum_{e<f} \bar{p}_{t,e}\bar{p}_{t,f}\langle\boldsymbol{\delta}_{t,e},\boldsymbol{\delta}_{t,f}\rangle. \end{aligned} \tag{21}$$

This identity applies to perturbations of complete expert outputs and does not linearize their internal nonlinearities. Cross terms vanish for Top-1 routing, but not necessarily for multiple co-activated experts. Their signs are unrestricted, so neither universal amplification nor beneficial cancellation is assumed. Updating expert e with all other expert factors fixed uses

$$\boldsymbol{r}_{t,e} = \boldsymbol{y}_t^\star - \boldsymbol{c}_t - \sum_{f \neq e} \bar{p}_{t,f} \widehat{f}_f(\widehat{\boldsymbol{x}}_t), \qquad \min_{\Theta_e} \frac{1}{T} \sum_t \|\boldsymbol{r}_{t,e} - \bar{p}_{t,e} \widehat{f}_e(\widehat{\boldsymbol{x}}_t; \Theta_e)\|_2^2. \tag{22}$$

A shared-factor update, in contrast, must account for every affected expert simultaneously. Experts not selected on a token contribute no gradient for that token. An expert never selected in the calibration set retains its initialized expert-specific factors in RCBF; no unseen-token accuracy guarantee follows.

Projected updates and sign-flip proposals. Let $\boldsymbol{J}_{e,m} = \partial \mathcal{L}_{\mathrm{R}} / \partial \widehat{\boldsymbol{V}}_e^{(m,1)}$ be obtained by differentiating the current MoE computation at its dequantized values. For fixed signs, the scale derivatives are

$$\frac{\partial \mathcal{L}_{\mathrm{R}}}{\partial s_i^r} = \sum_j J_{ij} B_{ij} s_j^c, \qquad \frac{\partial \mathcal{L}_{\mathrm{R}}}{\partial s_j^c} = \sum_i J_{ij} s_i^r B_{ij}. \tag{23}$$

Continuous proposals take a step along the corresponding gradient and are rounded with $\mathcal{Q}_{h,m}$ or $\mathcal{Q}_s$ before evaluating their deployed output. This uses derivatives of the continuous reconstruction at the current stored values; it does not claim an exact derivative through a discrete quantizer. Flipping one sign changes the factor entry by $-2 s_i^r B_{ij} s_j^c$, giving the first-order proposal score

$$\widehat{\Delta \mathcal{L}}_{ij} = -2 s_i^r B_{ij} s_j^c J_{ij}. \tag{24}$$

Negative scores prioritize possible improvements, but a sign flip is a finite perturbation, so the exact nonlinear loss decides acceptance. After an accepted update, affected outputs and proposal scores are refreshed. Several flips are not accepted by merely summing stale individual scores. Backtracking candidates are rounded and evaluated separately; if none lowers the loss, the current block is retained.

A closed-form down-factor subproblem. Hold intermediate activations, routing, and expert down factors fixed, and set

$$\boldsymbol{z}_t = \sum_e \bar{p}_{t,e} \widehat{\boldsymbol{V}}_e^{(\mathrm{down},1)} \boldsymbol{h}_{t,e}, \qquad \widetilde{\boldsymbol{y}}_t = \boldsymbol{y}_t^\star - \boldsymbol{c}_t - \sum_e \bar{p}_{t,e} \boldsymbol{b}_e^d. \tag{25}$$

Stack these as $\boldsymbol{Z} \in \mathbb{R}^{r_{\mathrm{down}} \times T}$ and $\boldsymbol{Y} \in \mathbb{R}^{d \times T}$. For the current down factor $\boldsymbol{A}_{\mathrm{old}}$ and $\lambda_{\mathrm{R}} > 0$, solve

$$\begin{aligned} \boldsymbol{A}_{\mathrm{cand}} &= \arg\min_{\boldsymbol{A}} \|\boldsymbol{Y} - \boldsymbol{A}\boldsymbol{Z}\|_F^2 + \lambda_{\mathrm{R}} \|\boldsymbol{A} - \boldsymbol{A}_{\mathrm{old}}\|_F^2 \\ &= (\boldsymbol{Y}\boldsymbol{Z}^\top + \lambda_{\mathrm{R}} \boldsymbol{A}_{\mathrm{old}})(\boldsymbol{Z}\boldsymbol{Z}^\top + \lambda_{\mathrm{R}} \boldsymbol{I})^{-1}. \end{aligned} \tag{26}$$

The normal equation follows by setting $2(\boldsymbol{A}\boldsymbol{Z} - \boldsymbol{Y})\boldsymbol{Z}^\top + 2\lambda_{\mathrm{R}}(\boldsymbol{A} - \boldsymbol{A}_{\mathrm{old}}) = 0$. Positive damping makes the system invertible even for rank-deficient $\boldsymbol{Z}$; a linear solve, rather than an explicit inverse, implements it. Only $\mathcal{Q}_{h,\mathrm{down}}(\boldsymbol{A}_{\mathrm{cand}})$ is tested for acceptance against the unregularized deployed loss in Eq. (20). The continuous minimizer alone is not a certificate of improvement after quantization. For gate/up factors, intermediate activations depend nonlinearly on the factor being changed, so Eq. (26) is not used as a closed-form solution. Accepted finite updates make the measured calibration loss decrease and remain nonnegative; this establishes monotonic objective values, not convergence to a global or stationary solution. A finite sweep limit is an explicit algorithm input.

A.1.2 Conditional Expert Residual Allocation

Frozen base representation and candidate generation. After RCBF, freeze all first-plane factors, ranks, scale formats, and shared gate/up factors. Let $\boldsymbol{A}_{m,0}$ denote the deployed shared factors at this transition; $\boldsymbol{A}_0 = \boldsymbol{A}_{\mathrm{down},0}$ remains the fixed reference for all subsequent down-factor readaptations. A residual target is fitted in these current coordinates, not taken from the original SVD factors:

$$\begin{aligned} \boldsymbol{F}_{e,m} &= \arg\min_{\boldsymbol{D}} \|\boldsymbol{W}_e^{(m)} - \boldsymbol{A}_{m,0}(\widehat{\boldsymbol{V}}_e^{(m,1)} + \boldsymbol{D})\|_F^2 + \xi \|\boldsymbol{D}\|_F^2 \\ &= (\boldsymbol{A}_{m,0}^\top \boldsymbol{A}_{m,0} + \xi \boldsymbol{I})^{-1} \boldsymbol{A}_{m,0}^\top (\boldsymbol{W}_e^{(m)} - \boldsymbol{A}_{m,0} \widehat{\boldsymbol{V}}_e^{(m,1)}), \qquad \xi > 0. \end{aligned} \tag{27}$$

This damped weight-space fit only constructs candidates; it is not CERA's final allocation objective. Fit a scaled second binary plane $\boldsymbol{T}_{e,m} = \operatorname{diag}(\boldsymbol{u}^r_{e,m})\boldsymbol{B}^{(2)}_{e,m}\operatorname{diag}(\boldsymbol{u}^c_{e,m})$ to $\boldsymbol{F}_{e,m}$ using Appendix A.3.6, including rounding of its retained scales. Partition the columns into fixed, nonempty, disjoint groups $\mathcal{J}_{m,g}$. For $a = (e, m, g)$, let $\boldsymbol{P}_{m,g}$ be the diagonal selector of those columns and define

$$\boldsymbol{R}_a = \boldsymbol{T}_{e,m}\boldsymbol{P}_{m,g}, \qquad \widehat{\boldsymbol{V}}^{(m)}_e(\mathcal{S}) = \widehat{\boldsymbol{V}}^{(m,1)}_e + \sum_{a\in\mathcal{S}_{e,m}} \boldsymbol{R}_a. \tag{28}$$

The candidate signs and scales are immutable during selection, including when the shared down factor changes. Each factor's second-plane row scales are shared across its selected groups; only selected columns require second-plane signs and column scales. The representation has at most two binary planes per factor entry, rather than an unconstrained number of residual planes.

Nonlinear candidate responses and shared-factor readaptation. Compute $\boldsymbol{h}_{t,e}(\mathcal{S})$ using the full expert nonlinearity and currently selected gate/up residuals. With the current down residuals, set

$$\boldsymbol{Z}_{\mathcal{S}}[:, t] = \sum_e \bar{p}_{t,e}\widehat{\boldsymbol{V}}^{(\mathrm{down})}_e(\mathcal{S})\boldsymbol{h}_{t,e}(\mathcal{S}). \tag{29}$$

The target $\boldsymbol{Y}$ subtracts the deployed fixed branches and original output-bias contributions exactly as in Eq. (25). Those terms are constant under the current fixed routing, even if teacher and student inputs differ. For any candidate set, consider the auxiliary continuous objective

$$\mathcal{J}(\boldsymbol{A}; \mathcal{S}) = \|\boldsymbol{Y} - \boldsymbol{A}\boldsymbol{Z}_{\mathcal{S}}\|^2_F + \lambda\|\boldsymbol{A} - \boldsymbol{A}_0\|^2_F, \qquad \lambda > 0. \tag{30}$$

Its derivative is $2(\boldsymbol{A}\boldsymbol{Z}_{\mathcal{S}} - \boldsymbol{Y})\boldsymbol{Z}^\top_{\mathcal{S}} + 2\lambda(\boldsymbol{A} - \boldsymbol{A}_0)$, giving

$$\boldsymbol{A}^\star_{\mathcal{S}} = (\boldsymbol{Y}\boldsymbol{Z}^\top_{\mathcal{S}} + \lambda\boldsymbol{A}_0)(\boldsymbol{Z}_{\mathcal{S}}\boldsymbol{Z}^\top_{\mathcal{S}} + \lambda\boldsymbol{I})^{-1}. \tag{31}$$

The down-factor system is solved anew or by a valid statistics update; its reference is always $\boldsymbol{A}_0$, not a silently moving reference. Only this shared output factor is readapted during allocation. No new high-precision correction matrix, per-token state, or trainable router is introduced at this stage.

Utility in the deployed state. Let $\mathcal{E}(\mathcal{S}; \boldsymbol{A}) = \|\boldsymbol{Y} - \boldsymbol{A}\boldsymbol{Z}_{\mathcal{S}}\|^2_F$. Under the same fixed targets and routing, this unnormalized error equals $T\mathcal{L}_{\mathrm{R}}$ for the corresponding representation. The search state is the pair $(\mathcal{S}, \boldsymbol{A}_{\mathcal{S}})$, initialized by $(\varnothing, \boldsymbol{A}_0)$. For $a \notin \mathcal{S}$, evaluate both the unchanged shared factor and the rounded continuous candidate:

$$\begin{aligned} \boldsymbol{A}_a &\in \arg\min_{\boldsymbol{A}\in\{\boldsymbol{A}_{\mathcal{S}},\, \mathcal{Q}_{h,\mathrm{down}}(\boldsymbol{A}^\star_{\mathcal{S}\cup\{a\}})\}} \mathcal{E}(\mathcal{S}\cup\{a\}; \boldsymbol{A}), \\ \Delta(a \mid \mathcal{S}) &= \mathcal{E}(\mathcal{S}; \boldsymbol{A}_{\mathcal{S}}) - \mathcal{E}(\mathcal{S}\cup\{a\}; \boldsymbol{A}_a). \end{aligned} \tag{32}$$

Ties retain the current shared factor. The notation $\Delta(a \mid \mathcal{S})$ implicitly includes the current deployed factor; it is not a claim that all histories reaching the same set have the same state. The unregularized deployed error, not $\min_{\boldsymbol{A}} \mathcal{J}(\boldsymbol{A}; \mathcal{S})$, determines the allocation gain. Let $\mathcal{M}(\mathcal{S})$ denote the occupied packed storage in bits, including the selected payloads, scales, descriptors, and array alignment, but excluding unused capacity reserved by fixed-size expert records. The storage-normalized utility is

$$u(a \mid \mathcal{S}) = \frac{\Delta(a \mid \mathcal{S})}{\Delta\mathcal{M}(a \mid \mathcal{S})}, \quad \Delta\mathcal{M}(a \mid \mathcal{S}) = \mathcal{M}(\mathcal{S}\cup\{a\}) - \mathcal{M}(\mathcal{S}). \tag{33}$$

The packed-group layout in Appendix A.2.2 gives positive occupied-payload increments for nonempty candidates. For compact records, this increment equals the serialized-size increment; for fixed-capacity records, it measures the capacity consumed by the candidate rather than the change in the padding-inclusive record size. A candidate is feasible only if it fits any per-expert record capacity and the complete serialized layer size satisfies $\mathcal{M}^{\mathrm{ser}}_l(\mathcal{S}\cup\{a\}) \le \mathcal{B}_l$, as defined in Eq. (56). Budgets include shared factors, first planes, fixed components, and any reserved record padding before allocation starts.

Exact finite gain and the limits of additivity. Define the current output residual and a candidate's exact output change at a fixed shared factor by

$$\boldsymbol{E}_{\mathcal{S}} = \boldsymbol{Y} - \boldsymbol{A}_{\mathcal{S}}\boldsymbol{Z}_{\mathcal{S}}, \qquad \boldsymbol{D}_a = \boldsymbol{A}_{\mathcal{S}}(\boldsymbol{Z}_{\mathcal{S}\cup\{a\}} - \boldsymbol{Z}_{\mathcal{S}}). \tag{34}$$

Expanding $\|\boldsymbol{E}_{\mathcal{S}} - \boldsymbol{D}_a\|_F^2$ gives the exact fixed-factor gain

$$\Delta_{\text{fixed}}(a \mid \mathcal{S}) = 2\langle \boldsymbol{E}_{\mathcal{S}}, \boldsymbol{D}_a \rangle - \|\boldsymbol{D}_a\|_F^2. \tag{35}$$

Because the unchanged factor is an option in Eq. (32), exact readaptation cannot give a smaller candidate gain than the fixed-factor option under the same evaluation. Nevertheless, either gain can be negative relative to the current state. For a fixed shared factor and additive output changes $\boldsymbol{D}_a$ that do not depend on the selected set, the restricted fixed-response model gives

$$\Delta_{\text{fixed}}(a \mid \mathcal{S}) = \Delta_{\text{fixed}}(a \mid \varnothing) - 2\sum_{b \in \mathcal{S}} \langle \boldsymbol{D}_a, \boldsymbol{D}_b \rangle. \tag{36}$$

Their inner products can have either sign, so this expression gives no general diminishing-returns guarantee. With nonlinear gate/up changes or a readapted shared factor, the fixed-response assumption does not hold in general, and Eq. (36) is not used as an exact update rule. Equation (35) remains exact when the complete current candidate response is recomputed.

Screening, refreshing, and stopping. At each state, rank feasible candidates by a cached or recomputed fixed-factor gain-per-bit score, then evaluate a shortlist with the deployed-factor test in Eq. (32). A nonpositive screening score does not prove that readaptation cannot make the candidate useful. If the shortlist has no positive exact gain, enlarge it through the remaining feasible candidates before declaring no improving candidate at that state. Otherwise, commit the highest positive feasible utility within the evaluated shortlist, update the shared down factor, and refresh affected responses and all utility estimates before the next selection. This is not an exhaustive best-utility search unless all feasible candidates are evaluated. With a strict numerical acceptance tolerance $\varepsilon_{\text{acc}} > 0$, accept only $\Delta > \varepsilon_{\text{acc}}$. Sets grow without removing selected groups and contain finitely many candidates; the search terminates after at most that many accepted additions, or earlier under an explicit evaluation cap. Stopping because of a cap is reported separately from exhausting improving candidates. Neither monotonic calibration-error decrease nor finite termination implies optimal accuracy, a globally optimal allocation, or performance on unseen data.

Response caching and computational scope. Only tokens routed to expert e require recomputing that expert for candidate (e, m, g). For a down candidate with all intermediate activations fixed, let $\boldsymbol{H}_e$ contain its current intermediate activations and let $\boldsymbol{P}_e = \operatorname{diag}(\bar{p}_{1,e}, \ldots, \bar{p}_{T,e})$. Then $\Delta \boldsymbol{Z} = \boldsymbol{R}_a \boldsymbol{H}_e \boldsymbol{P}_e$, with rank at most $|\mathcal{J}_{m,g}|$. Writing $\boldsymbol{Z}' = \boldsymbol{Z} + \Delta \boldsymbol{Z}$, exact statistic updates are

$$\begin{aligned} \boldsymbol{Z}'\boldsymbol{Z}'^{\top} &= \boldsymbol{Z}\boldsymbol{Z}^{\top} + \boldsymbol{Z}\Delta\boldsymbol{Z}^{\top} + \Delta\boldsymbol{Z}\boldsymbol{Z}^{\top} + \Delta\boldsymbol{Z}\Delta\boldsymbol{Z}^{\top}, \\ \boldsymbol{Y}\boldsymbol{Z}'^{\top} &= \boldsymbol{Y}\boldsymbol{Z}^{\top} + \boldsymbol{Y}\Delta\boldsymbol{Z}^{\top}. \end{aligned} \tag{37}$$

Gate/up candidates instead require recomputing the actual nonlinear $\boldsymbol{h}_{t,e}$; no equally small-rank update is assumed. For dense statistics, one shared-factor solve costs $O(r_d^3 + dr_d^2)$ after forming $\boldsymbol{Z}\boldsymbol{Z}^{\top}$ and $\boldsymbol{Y}\boldsymbol{Z}^{\top}$, where $r_d = r_{\text{down}}$. Forming those statistics costs $O(Tr_d^2 + dTr_d)$ without exploiting sparsity or incremental updates. Full candidate-response caches are not required: compute expert-local or shortlisted responses on demand and discard rejected temporary buffers. These operation counts characterize the stated screening and caching procedure; they do not determine wall-clock calibration time or peak memory.

A.1.3 Input-Aware Margin Preservation in MOBR

Teacher targets and two sources of logit error. Use the column-vector convention $\boldsymbol{z}_t^{\star} = \boldsymbol{G}^{\top}\boldsymbol{x}_t^{\star}$ and $\widehat{\boldsymbol{z}}_t = \widehat{\boldsymbol{G}}^{\top}\widehat{\boldsymbol{x}}_t$ for $\boldsymbol{G} \in \mathbb{R}^{d \times E}$. The exact error decomposition is

$$\widehat{\boldsymbol{z}}_t - \boldsymbol{z}_t^{\star} = (\widehat{\boldsymbol{G}} - \boldsymbol{G})^{\top}\widehat{\boldsymbol{x}}_t + \boldsymbol{G}^{\top}(\widehat{\boldsymbol{x}}_t - \boldsymbol{x}_t^{\star}). \tag{38}$$

The teacher logits, selected set $S_t^{\star}$, and pair set $\mathcal{P}_t = \{(i,j) : i \in S_t^{\star},\ j \notin S_t^{\star}\}$ are cached and remain fixed during router optimization. For standard global Top-k selection, strictly positive student margins on every pair in $\mathcal{P}_t$ imply the same selected set, provided the teacher set is well defined. If the smallest teacher margin is $\gamma_t > 0$, the sufficient condition $\|\widehat{\boldsymbol{z}}_t - \boldsymbol{z}_t^{\star}\|_{\infty} < \gamma_t/2$ preserves every selected–unselected ordering. This condition concerns total logit error, including inherited input drift; it is not guaranteed by weight reconstruction alone. Ties require the checkpoint's deterministic tie-breaking rule and are excluded from this strict-margin sufficient condition.

Pairwise KL and derivatives at the student iterate. For a teacher probability $p = \sigma(m^\star/\tau)$ and variable student margin m, set $q(m) = \sigma(m/\tau)$ with $\tau > 0$. The pairwise divergence is

$$D(m) = p \log \frac{p}{q(m)} + (1-p) \log \frac{1-p}{1-q(m)}. \tag{39}$$

Using $q'(m) = q(m)(1-q(m))/\tau$ gives

$$D'(m) = \frac{q(m)-p}{\tau}, \qquad D''(m) = \frac{q(m)(1-q(m))}{\tau^2}. \tag{40}$$

For stable computation, the variable part of D is $\mathrm{softplus}(m/\tau) - p\, m/\tau$; the omitted target-only entropy is identical for all candidates. At a current student router $\boldsymbol{G}^{(0)}$, write $q^{(0)}_{t,ij}$ for its probabilities and define

$$\beta_{t,ij} = \frac{q^{(0)}_{t,ij} - q^\star_{t,ij}}{\tau}, \qquad \omega_{t,ij} = \frac{q^{(0)}_{t,ij}(1-q^{(0)}_{t,ij})}{\tau^2}. \tag{41}$$

The local expansion is $D(m^{(0)} + \delta m) = D(m^{(0)}) + \beta \delta m + \frac{1}{2}\omega(\delta m)^2 + o((\delta m)^2)$. The linear term vanishes only when the student and teacher pair probabilities coincide. With perturbed inputs, setting $\boldsymbol{G}^{(0)} = \boldsymbol{G}$ does not generally make that term zero.

Full weight-space gradient and curvature. Let $\boldsymbol{a}_{ij} = \boldsymbol{e}_i - \boldsymbol{e}_j$ and vectorize router matrices columnwise. For $\delta \boldsymbol{G}$, the margin change at fixed student inputs is

$$\delta m_{t,ij} = \boldsymbol{a}_{ij}^\top \delta \boldsymbol{G}^\top \widehat{\boldsymbol{x}}_t = (\boldsymbol{a}_{ij} \otimes \widehat{\boldsymbol{x}}_t)^\top \mathrm{vec}(\delta \boldsymbol{G}). \tag{42}$$

Averaging over tokens and summing all pairs yields

$$\begin{aligned} \boldsymbol{g} &= \frac{1}{T} \sum_t \sum_{(i,j)\in\mathcal{P}_t} \beta_{t,ij} (\boldsymbol{a}_{ij} \otimes \widehat{\boldsymbol{x}}_t), \\ \boldsymbol{H} &= \frac{1}{T} \sum_t \boldsymbol{C}_t \otimes (\widehat{\boldsymbol{x}}_t \widehat{\boldsymbol{x}}_t^\top), \qquad \boldsymbol{C}_t = \sum_{(i,j)\in\mathcal{P}_t} \omega_{t,ij} \boldsymbol{a}_{ij} \boldsymbol{a}_{ij}^\top. \end{aligned} \tag{43}$$

Here $\boldsymbol{g} \in \mathbb{R}^{dE}$ and $\boldsymbol{H} \in \mathbb{R}^{dE \times dE}$ are the exact derivatives of the linear-router pairwise KL at the current student point. The local surrogate is $\boldsymbol{g}^\top \mathrm{vec}(\delta \boldsymbol{G}) + \frac{1}{2} \mathrm{vec}(\delta \boldsymbol{G})^\top \boldsymbol{H}\, \mathrm{vec}(\delta \boldsymbol{G})$. Each $\boldsymbol{C}_t$ is positive semidefinite and annihilates the all-ones vector: a common shift of all expert logits leaves pairwise margins unchanged. Consequently, invertibility is not guaranteed without damping even when input features span their space. The off-diagonal blocks encode genuine competition and are not mathematically zero.

Damped block approximation. For each expert column e, define

$$\begin{aligned} b_{t,e} &= \sum_{(i,j)\in\mathcal{P}_t} \beta_{t,ij} (\boldsymbol{a}_{ij})_e, \quad v_{t,e} = \sum_{(i,j)\in\mathcal{P}_t} \omega_{t,ij} (\boldsymbol{a}_{ij})_e^2, \\ \boldsymbol{g}_e &= \frac{1}{T} \sum_t b_{t,e} \widehat{\boldsymbol{x}}_t, \qquad \boldsymbol{H}_e = \frac{1}{T} \sum_t v_{t,e} \widehat{\boldsymbol{x}}_t \widehat{\boldsymbol{x}}_t^\top + \rho \boldsymbol{I}. \end{aligned} \tag{44}$$

The undamped terms are the diagonal blocks of Eq. (43); $\rho \boldsymbol{I}$, with $\rho > 0$, is a stabilizing surrogate term rather than part of the original KL Hessian. For a selected expert, $v_{t,e}$ sums curvatures against unselected experts; for an unselected expert, it sums curvatures against selected experts. Replacing $\boldsymbol{H}$ with these diagonal blocks discards cross-column terms but retains the true first derivative. The resulting surrogate is

$$\mathcal{Q}(\{\boldsymbol{\delta}_e\}) = \sum_e \left[\boldsymbol{g}_e^\top \boldsymbol{\delta}_e + \frac{1}{2} \boldsymbol{\delta}_e^\top \boldsymbol{H}_e \boldsymbol{\delta}_e \right]. \tag{45}$$

If both inputs and router parameters match the teacher, then $\beta = 0$ and the teacher-centered curvature is recovered as a special case. No full-coupled versus block-diagonal performance claim is inferred from this derivation.

Binary-constrained column updates. Use $\widehat{\boldsymbol{G}}_{:e} = \alpha_e \boldsymbol{b}_e$, where $\boldsymbol{b}_e \in \{-1, +1\}^d$ and $\alpha_e \geq 0$. Let $\boldsymbol{w}_e^{(0)} = \boldsymbol{G}_{:e}^{(0)}$ and $\boldsymbol{t}_e = \boldsymbol{H}_e \boldsymbol{w}_e^{(0)} - \boldsymbol{g}_e$. For fixed $\boldsymbol{b}_e$, substitution into Eq. (45) gives a one-dimensional quadratic with minimizer

$$\alpha_e^\star(\boldsymbol{b}_e) = \max\left(0, \frac{\boldsymbol{b}_e^\top \boldsymbol{t}_e}{\boldsymbol{b}_e^\top \boldsymbol{H}_e \boldsymbol{b}_e}\right). \tag{46}$$

The denominator is strictly positive because of $\rho > 0$; no extra denominator perturbation is needed to state this exact continuous scale solution. The term $-\boldsymbol{g}_e$ is essential and is absent only in the zero-gradient special case. A projected-Newton proposal uses $\boldsymbol{b}_e = \operatorname{sign}(\boldsymbol{w}_e^{(0)} - \boldsymbol{H}_e^{-1}\boldsymbol{g}_e)$, with zero entries mapped to $+1$, followed by Eq. (46). This is a proposal, not the global minimizer over signs. For optional coordinate flips at fixed α, let $\boldsymbol{h} = \boldsymbol{g}_e + \boldsymbol{H}_e(\alpha \boldsymbol{b} - \boldsymbol{w}_e^{(0)})$. Flipping sign i changes the quadratic surrogate by

$$\Delta \mathcal{Q}_i = -2\alpha b_i h_i + 2\alpha^2 [\boldsymbol{H}_e]_{ii}. \tag{47}$$

After selecting a sign proposal, refit and round its scale to the retained format. Accept a column or joint proposal only if the exact full pairwise KL decreases on the same calibration set. Recompute student probabilities and local derivatives after a committed update. Initialization is already binary; the procedure does not require the first lossy binarization to improve on the full-precision teacher.

Routing adapters, guarantees, and calibration cost. The formulas above assume affine expert logits and standard selected–unselected comparisons. Fixed additive routing biases can be included in the margins; their derivatives with respect to $\boldsymbol{G}$ are unchanged, and retained bias storage must be counted. A shared monotone score transform preserves rankings but not necessarily the numerical margins or aggregation weights. Grouped selection, expert-specific score transformations, capacity constraints, and special dispatch rules must use the checkpoint's actual selection adapter; the global Top-k sufficient condition is not silently applied to a different router. For the evaluated OLMoE, DeepSeek-V2-Lite, and Qwen3 checkpoints, selection uses global Top-k scores; OLMoE and DeepSeek retain the selected softmax weights without renormalization, whereas Qwen3 renormalizes them over the selected experts (Allen Institute for AI, 2025; DeepSeek-AI, 2024a; Qwen Team, 2025). After optimization, rerun the actual deployed router to cache $\bar{p}_{t,e}$, not its auxiliary Bernoulli probabilities. For standard Top-k, there are $k(E-k)$ teacher pairs per token. Explicit storage of the full Hessian costs $O(d^2E^2)$, versus $O(Ed^2)$ for all diagonal blocks; blocks can instead be accumulated in bounded batches. Naive dense block construction costs $O(TEd^2)$ after pair coefficients are available, and pair evaluation costs $O(Tk(E-k))$ in addition to computing logits. These are arithmetic/storage bounds, not measured calibration times.

A.2 Deployment Storage and Effective Bitwidth

Storage is counted from the final serialized representation, not from the number of optimization variables or the nominal bitwidth of its sign matrices. We distinguish all-expert resident weight storage, per-token active encoded payload, and total runtime memory. The formulas below separate occupied payload from padding-inclusive record storage; Appendix A.4.2 instantiates them for the evaluated models.

A.2.1 Fixed Representation and Resident Storage

Original parameter counts and factor shapes. For one layer, the original routed projection count is $P_l = \sum_m E_l O_{l,m} I_{l,m}$. The number of entries in the shared factors and first-plane coefficient arrays is

$$N_l^{\text{factor}} = \sum_m \big(O_{l,m} r_{l,m} + E_l r_{l,m} I_{l,m}\big). \tag{48}$$

This coefficient-entry count is not the number of unconstrained trainable parameters and excludes retained scales and metadata. For one projection, the ratio to EOI is

$$\frac{Or + ErI}{EOI} = \frac{r}{EI} + \frac{r}{O}. \tag{49}$$

At $r = O$, the count increases by O^2, with relative increase $O/(EI)$. There is no universal "below one percent" bound without specifying the dimensions and expert count. For general retained rank, count reduction requires $r < EOI/(O + EI)$. Byte-level compression can still occur when the coefficient-entry count increases because most expert coefficients are stored as signs.

Packing convention and fixed payload. Let $\mathrm{pack}(n) = b_{\mathrm{align}}\lceil n/b_{\mathrm{align}}\rceil$ bits for an alignment unit of b_{align} bits, with $\mathrm{pack}(0) = 0$. A byte-packed implementation has at least $b_{\mathrm{align}} = 8$; any larger buffer alignment must use its actual value. For a shared factor, write

$$S^A_{l,m} = \mathrm{pack}(B_{h,l,m}O_{l,m}r_{l,m}) + H^A_{l,m}, \tag{50}$$

where H^A includes its stored quantizer scales, zero points if used, descriptors, and their alignment. For compact notation, assume both base row and column scales use B_s bits; different formats replace this sum by their actual separate costs. For a deterministic column partition with $G_{l,m}$ groups and a fixed stored group bitmap, the base factor payload is

$$S^{(1)}_{l,e,m} = \mathrm{pack}(r_{l,m}I_{l,m}) + \mathrm{pack}(B_s(r_{l,m} + I_{l,m})) + \mathrm{pack}(G_{l,m}) + H^{(1)}_{l,e,m}. \tag{51}$$

The term $H^{(1)}$ includes remaining factor descriptors and alignment costs not counted elsewhere. The bitmap stores selected residual groups, including an all-zero bitmap when no residual is selected. A serializer using sparse indices or conditional bitmaps must substitute its actual metadata cost rather than add both layouts. The same byte must never be charged twice under a payload and a metadata term.

A.2.2 Residual Payloads and Incremental Costs

Let $\mathcal{G}_{l,e,m}(\mathcal{S})$ be the selected groups for one factor, let $c_g = |\mathcal{J}_{m,g}|$, and let $n_{l,e,m}(\mathcal{S}) = \sum_{g\in\mathcal{G}_{l,e,m}(\mathcal{S})} c_g$. An explicit layout stores second-plane row scales once per enabled factor and signs/column scales in independent selected-group buffers:

$$\begin{aligned} S^{(2)}_{l,e,m}(\mathcal{S}) = {} & \mathbf{1}\{n_{l,e,m}(\mathcal{S}) > 0\}\,\mathrm{pack}(B_s r_{l,m}) \\ & + \sum_{g\in\mathcal{G}_{l,e,m}(\mathcal{S})} \Big[\mathrm{pack}(r_{l,m}c_g) + \mathrm{pack}(B_s c_g) + H^g_{l,e,m}\Big]. \end{aligned} \tag{52}$$

Any group offsets or per-group headers are counted in H^g; deterministic traversal may make this term zero. This layout counts each selected group once and shares its second-plane row scales across groups of the same factor. For the proposed candidate $a = (e, m, g)$, its increment under this layout is

$$\begin{aligned} \Delta\mathcal{M}_l(a \mid \mathcal{S}) = {} & \mathrm{pack}(r_{l,m}c_g) + \mathrm{pack}(B_s c_g) + H^g_{l,e,m} \\ & + \mathbf{1}\{n_{l,e,m}(\mathcal{S}) = 0\}\,\mathrm{pack}(B_s r_{l,m}). \end{aligned} \tag{53}$$

The first selected group pays for shared second-plane row scales, so cost depends on the current set even before considering reconstruction interactions. Before array alignment and record descriptors, the coefficient payload of one factor is

$$\begin{aligned} S^{\mathrm{coef}}_{e,m}(\mathcal{S}) = {} & r_m I_m + B_s(r_m + I_m) + G_m \\ & + r_m n_{e,m}(\mathcal{S}) + B_s n_{e,m}(\mathcal{S}) + \mathbf{1}\{n_{e,m}(\mathcal{S}) > 0\}B_s r_m. \end{aligned} \tag{54}$$

Budget enforcement uses the packed costs in Eqs. (51) and (52), rather than this unpadded expression. The occupied packed storage of each layer is

$$\mathcal{M}_l(\mathcal{S}) = S^{\mathrm{fixed}}_l + \sum_m S^A_{l,m} + \sum_{e,m}\big(S^{(1)}_{l,e,m} + S^{(2)}_{l,e,m}(\mathcal{S})\big). \tag{55}$$

The fixed term contains the deployed router, retained biases, always-active experts, and any other resident components assigned to this layer budget. A component charged here is excluded from any later outside-budget total. Because readaptation preserves factor shape, precision, and metadata layout, changing shared-factor values does not change its storage cost.

Occupied payload and fixed-capacity records. Let $Q^{\mathrm{occ}}_{l,e}(\mathcal{S}) = \sum_m [S^{(1)}_{l,e,m} + S^{(2)}_{l,e,m}(\mathcal{S})]$ be one expert's occupied payload in bits. For a fixed expert-record capacity $C_{l,e}$ in bytes, feasibility requires $Q^{\mathrm{occ}}_{l,e}(\mathcal{S}) \le 8C_{l,e}$ for every expert. The complete serialized layer size is

$$\begin{aligned} \mathcal{M}^{\mathrm{ser}}_l(\mathcal{S}) &= \mathcal{M}_l(\mathcal{S}) + \mathcal{M}^{\mathrm{pad}}_l(\mathcal{S}), \\ \mathcal{M}^{\mathrm{pad}}_l(\mathcal{S}) &= \sum_e \left[8C_{l,e} - Q^{\mathrm{occ}}_{l,e}(\mathcal{S})\right]. \end{aligned} \tag{56}$$

For compact records without reserved capacity, $\mathcal{M}_l^{\text{pad}} = 0$ instead. Under fixed capacities, accepting a residual consumes occupied payload and reduces unused padding by the same number of bits; the total serialized size remains fixed. Thus, Eq. (33) divides by the positive occupied-payload increment, not by the zero increment of the fixed serialized record. All reported BPWs include the full serialized size, including unused capacity. The initial configuration must satisfy $\mathcal{M}_l^{\text{ser}}(\varnothing) \le \mathcal{B}_l$ and all per-expert capacities; otherwise it is infeasible without changing the configuration before calibration.

A.2.3 Active Payload, Resident Weights, and Runtime Memory

For a single routed projection, let $Q_e(\mathcal{S})$ denote its expert-specific encoded payload, including any assigned record padding, and let S^A denote its shared-factor payload. For compact storage, $Q_e = S_e^{(1)} + S_e^{(2)}$. When several projections share one fixed expert record, the same formulas apply to their combined record, with shared-factor costs summed across projections. All-expert resident storage and the ideal selected encoded payload for token t are

$$S_{\text{resident}} = S^A + \sum_{e=1}^{E} Q_e(\mathcal{S}), \qquad S_{\text{active},t} = S^A + \sum_{e \in S_t} Q_e(\mathcal{S}). \tag{57}$$

The second expression assumes the shared factor is accessed once for that projection and counts the selected buffers; it is not total on-device memory. Inactive experts remain resident in the deployment setting considered here. Actual memory traffic depends on batching, buffer reuse, caches, dequantization, and kernel scheduling, and is not determined by Eq. (57) alone. For a batch, the union of selected experts is relevant to ideal reusable payload; it need not contain exactly batch-size times k distinct experts.

For uniform expert payload $Q_e = Q$ and a fixed Top-k, dividing by the original target weight counts gives

$$\overline{B}_{\text{static}} = \frac{S^A + EQ}{EOI}, \qquad \overline{B}_{\text{active}} = \frac{S^A + kQ}{kOI}. \tag{58}$$

Thus, if $E > k$ and $S^A > 0$,

$$\overline{B}_{\text{active}} - \overline{B}_{\text{static}} = \frac{S^A}{OI}\left(\frac{1}{k} - \frac{1}{E}\right) > 0. \tag{59}$$

Shared-factor overhead is amortized less, not more, on a per-token active basis. With heterogeneous CERA allocations, active experts have different Q_e, and their activation-weighted mean need not equal the all-expert mean. No unconditional active-versus-static bitwidth ordering is then asserted.

Total peak runtime memory additionally includes activations, KV cache, workspaces, allocator effects, and temporary dequantized buffers. Calibration-only teacher outputs, covariance matrices, candidate planes not selected, gradients, and optimizer state do not belong to deployment weight storage, but their cost must be included in calibration resource measurements. Reducing encoded weight storage does not by itself establish a particular inference speedup.

A.2.4 Whole-Model Bitwidth and Comparison Scope

Let $\mathcal{T}$ be a common set of original target weights, $P_{\mathcal{T}}$ their original count, and $S_{\mathcal{T}}$ the final stored bits for their replacement representation, including method-specific scales, shared factors, residuals, metadata, and reserved padding. Let $P_{\mathcal{U}}$ and $S_{\mathcal{U}}$ describe all remaining original parameters and their retained storage. Define

$$\text{BPW}_{\text{target}} = \frac{S_{\mathcal{T}}}{P_{\mathcal{T}}}, \qquad \text{BPW}_{\text{whole}} = \frac{S_{\mathcal{T}} + S_{\mathcal{U}}}{P_{\mathcal{T}} + P_{\mathcal{U}}}. \tag{60}$$

The denominators use original model weights, not the number of factor entries after reparameterization. If all non-target parameters are retained in BF16, their numerical tensor payload is $16P_{\mathcal{U}}$ bits; actual metadata and alignment are still added. Quantized routers and any attention-weight quantization are counted using their own actual representation, not assigned the expert-factor formula automatically. For a one-scale-per-column binary router, the unpadded payload is $dE + B_s E$ bits before its retained biases and metadata. All-expert resident comparisons use the same $\mathcal{T}$, the same

Table 5: Storage accounting scopes for one routed projection. Q_e includes base and selected residual payloads, scales, and metadata; S^A includes the shared-factor quantizer metadata. Reserved record padding is included in the encoded payload.

Quantity	Full-precision reference	Factorized binary representation
Coefficient entries	EOI	$Or + ErI$ before scales/residuals
Resident projection bits	$B_{\mathrm{fp}}EOI$ plus metadata	$S^A + \sum_e Q_e$
Selected token payload	$B_{\mathrm{fp}}kOI$ plus accessed metadata	$S^A + \sum_{e \in S_t} Q_e$
Target resident BPW	Reference bits divided by EOI	$(S^A + \sum_e Q_e)/(EOI)$
Runtime memory	Weights plus runtime states	Resident weights plus runtime states

non-target scope, and consistent counting of each baseline's own metadata. No baseline's bitmap or residual layout is inferred solely from its nominal weight bitwidth.

For a uniform full-precision reference with B_{fp} bits per weight, the compressed-to-reference storage fraction is

$$\mathrm{CF}_{\mathrm{resident}} = \frac{S_{\mathcal{T}} + S_{\mathcal{U}}}{B_{\mathrm{fp}}(P_{\mathcal{T}} + P_{\mathcal{U}}) + H_{\mathrm{fp}}}, \tag{61}$$

where H_{fp} is any baseline overhead included in the chosen serialized-storage scope. Smaller fractions mean greater compression; a fold compression ratio is the reciprocal and must be named separately. Per-token active fractions instead use the corresponding active reference and do not replace Eq. (61). For heterogeneous precision or actual checkpoint storage, use the exact reference bytes rather than assuming a uniform BF16 payload.

For reporting, distinguish decimal GB (10^9 bytes) from GiB (2^{30} bytes) and divide bit totals by eight before either conversion. Whole-model resident storage is $S_{\mathrm{outside}} + \sum_l \mathcal{M}_l^{\mathrm{ser}}(\mathcal{S}_l)$, with all terms in bits and every stored component counted exactly once. A matched-storage comparison requires matching this actual total or a clearly specified common target scope; equal nominal sign bitwidths alone are insufficient.

A.3 Layerwise Calibration and Pseudocode

Algorithms 1–6 expand the main-text procedure. Their execution order is MOBR $\rightarrow$ RCBF $\rightarrow$ CERA for each layer, although the theoretical subsections follow the main text's representation–allocation–routing exposition. The operator Route_l is the original model-specific selection and aggregation routine with only its targeted router weights replaced. All acceptance tests use the same fixed calibration examples within a layer. Teacher caches can be generated per layer or held off device; the schedule does not require all layers' teacher states to reside on the GPU simultaneously. The original FP teacher is never modified.

A.3.1 Overall Layerwise Workflow

Algorithm 1: BitMoE: Layerwise Calibration with Fixed Deployed Routing

Input: FP teacher $\mathcal{M}^\star$; calibration sequences $\mathcal{D}$; fixed ranks, layer budgets, expert-record capacities, quantizers, non-expert quantization policy, model routing adapters, and solver controls

Output: Deployed model $\widehat{\mathcal{M}}$

Cache paired teacher states, MoE outputs, and original router targets on $\mathcal{D}$;
Initialize student storage and finalize preceding non-expert operations according to the given policy;
for *each MoE layer l in forward order* **do**
  Obtain $\widehat{\boldsymbol{X}}_l$ by executing the actual finalized student prefix on $\mathcal{D}$;
  Retrieve teacher $\boldsymbol{X}_l^\star$, $\boldsymbol{Y}_l^\star$, and router targets for the same tokens;
  Compute and freeze the current student always-active branch contributions;
  $\widehat{\boldsymbol{G}}_l \leftarrow \mathrm{MOBR}(\boldsymbol{G}_l, \boldsymbol{X}_l^\star, \widehat{\boldsymbol{X}}_l)$ using Algorithm 2;
  $\bar{p}_{t,e} \leftarrow \mathrm{Route}_l(\widehat{\boldsymbol{G}}_l, \widehat{\boldsymbol{x}}_{l,t})_e$ for all calibration tokens;
  Freeze the deployed router and $\{\bar{p}_{t,e}\}$ for this layer;
  $\Theta_l^{(1)} \leftarrow \mathrm{RCBF}(\{\boldsymbol{W}_{l,e}^{(m)}\}, \widehat{\boldsymbol{X}}_l, \boldsymbol{Y}_l^\star, \bar{p})$ using Algorithm 3;
  if *the initial serialized layer size or an expert's base record exceeds its budget* **then**
    Report an infeasible configuration and stop without changing ranks;
  $\mathcal{C}_l \leftarrow \mathrm{BuildCandidates}(\Theta_l^{(1)}, \{\boldsymbol{W}_{l,e}^{(m)}\})$ using Algorithm 4;
  $(\mathcal{S}_l, \boldsymbol{A}_{l,\mathrm{down}}) \leftarrow \mathrm{CERA}(\Theta_l^{(1)}, \mathcal{C}_l, \mathcal{B}_l)$ using Algorithm 5;
  Serialize shared factors, first planes, selected residuals, scales, router, metadata, and fixed-record padding;
  Verify the exact stored cost and execute this finalized layer in its deployment format;
  Advance through intervening fixed/finalized operations to generate the next layer's inputs;
  Release obsolete layer-local teacher and candidate caches;
Return $\widehat{\mathcal{M}}$ with all retained non-target components and no calibration-only tensors;

A.3.2 MOBR Router Calibration

Algorithm 2: MOBR with Student-Point Gradient and Curvature

Input: Teacher router $\boldsymbol{G}$; paired $\boldsymbol{X}^\star$, $\widehat{\boldsymbol{X}}$; fixed teacher pair sets; temperature τ; damping ρ; retained scale format; iteration/proposal limits and acceptance tolerance

Output: Binary code-and-scale router $\widehat{\boldsymbol{G}}$

Cache $q_{t,ij}^\star$ from teacher logits and the model's teacher selection adapter;
Initialize $\boldsymbol{b}_e = \mathrm{sign}(\boldsymbol{G}_{:e})$, mapping zero to $+1$;
Initialize $\alpha_e = \mathcal{Q}_s(\|\boldsymbol{G}_{:e}\|_1/d)$ and $\boldsymbol{G}_{:e}^{(0)} = \alpha_e \boldsymbol{b}_e$;
Compute the exact pairwise KL for this binary initial router;
for *each allowed optimization sweep* **do**
  changed $\leftarrow$ false;
  for *each router column e* **do**
    Evaluate current student probabilities on $\widehat{\boldsymbol{X}}$;
    Accumulate $\boldsymbol{g}_e$ and $\boldsymbol{H}_e$ using Eq. (44);
    Form the projected-Newton sign proposal and screened coordinate-flip proposals;
    Include the current signs among the proposals;
    for *each proposed sign vector $\boldsymbol{b}$* **do**
      Compute $\alpha^\star(\boldsymbol{b})$ using Eq. (46);
      Round its scale and evaluate the full pairwise KL with only this column replaced;
    if *a deployed candidate strictly lowers the exact KL by the acceptance tolerance* **then**
      Commit the lowest-KL evaluated candidate;
      Refresh affected logits and set changed $\leftarrow$ true;
  if changed $=$ false **then**
    stop;
Return the current binary signs and retained column scales;

The pair targets are fixed, but the student probabilities and curvatures are refreshed after commits. This algorithm does not apply MOBR curvature to expert factor matrices.

A.3.3 RCBF BASE-REPRESENTATION REFINEMENT

Algorithm 3: RCBF: Binary-Aware Factor Refinement under Fixed Routing

Input: Original projection matrices; teacher targets; fixed student inputs, branches, and routing; ranks; quantizers; damping λ_{R}; steps, sweep limits, sign-proposal controls, and acceptance tolerance
Output: Shared factors and one-plane binary expert factors $\Theta^{(1)}$

for *each projection type* m **do**
  Initialize $\boldsymbol{A}_m^{\mathrm{svd}}$ and $\{\boldsymbol{V}_e^{(m,0)}\}$ using Eq. (15);
  $\boldsymbol{A}_m \leftarrow \mathcal{Q}_{h,m}(\boldsymbol{A}_m^{\mathrm{svd}})$;
  Fit and round each one-plane expert factor using Algorithm 6;
Evaluate and cache the actual nonlinear aggregate output and $\mathcal{L}_{\mathrm{R}}$;
for *each allowed sweep* **do**
  changed $\leftarrow$ false;
  for *each shared factor or scale block* **do**
    Hold other blocks fixed and differentiate Eq. (20) through the current layer;
    Form backtracking projected-gradient proposals;
    For the shared down factor, also form the fixed-activation ridge proposal in Eq. (26);
    Round each candidate to its deployment format and evaluate actual aggregate error;
    Commit a candidate only if its loss decrease exceeds the tolerance; refresh caches and mark changed;
  for *each expert factor in the sign-update schedule* **do**
    Differentiate the current aggregate loss with respect to the factor;
    Rank sign flips using Eq. (24);
    Test a bounded set of finite flips through the actual nonlinear expert computation;
    Commit only a loss decrease exceeding the tolerance; mark changed and refresh before the next proposal;
  if changed $=$ false **then**
    stop;
Return the stored shared factors, binary signs, and retained first-plane scales;

All changes are evaluated against the same aggregate objective, including shared-factor proposals affecting multiple experts. The algorithm uses no residual-selection mask from a future stage.

A.3.4 CONSTRUCTION OF FIXED RESIDUAL CANDIDATES

Algorithm 4: CERA Candidate Construction in the Current Shared Coordinates

Input: RCBF factors; original projection matrices; damping ξ; fixed column groups; binary-fit controls and retained scale format
Output: Immutable candidate set $\mathcal{C} = \{(e, m, g, \boldsymbol{R}_a)\}$ and its storage descriptors

Freeze the first planes, ranks, and shared gate/up factors;
Save $\boldsymbol{A}_{m,0}$ for every projection and $\boldsymbol{A}_0 = \boldsymbol{A}_{\mathrm{down},0}$;
$\mathcal{C} \leftarrow \varnothing$;
for *each expert* e *and projection* m **do**
  Solve for $\boldsymbol{F}_{e,m}$ using Eq. (27) with $\boldsymbol{A}_{m,0}$;
  Fit one scaled binary plane to $\boldsymbol{F}_{e,m}$ using Algorithm 6;
  Freeze that plane's signs and rounded row/column scales;
  for *each predetermined nonempty column group* g **do**
    Define $\boldsymbol{R}_{(e,m,g)}$ by restricting this plane to group g;
    Register its signs, selected column scales, shared second-row-scale identifier, and packing descriptor;
    Add the immutable candidate to $\mathcal{C}$;
Return $\mathcal{C}$; store candidate descriptions lazily if necessary;

Group restriction can be represented by views or indices during calibration; a full zero-padded matrix need not be materialized. Rejected or unselected candidate planes are not serialized into the deployed model.

A.3.5 CONDITIONAL RESIDUAL SELECTION AND READAPTATION

Algorithm 5: CERA: Budgeted Conditional Selection with Deployed-Error Checks

Input: RCBF state; immutable candidates $\mathcal{C}$; layer budget $\mathcal{B}_l$ and expert-record capacities; occupied/serialized storage evaluators; fixed student inputs, branches, routing, and targets; λ; shortlist width, evaluation cap, and $\varepsilon_{\mathrm{acc}}$

Output: Selected set $\mathcal{S}$ and deployed shared down factor $\boldsymbol{A}_{\mathcal{S}}$

$\mathcal{S} \leftarrow \varnothing$, $\boldsymbol{A}_{\mathcal{S}} \leftarrow \boldsymbol{A}_0$;
Compute $\boldsymbol{Y}$, $\boldsymbol{Z}_{\mathcal{S}}$ and the current deployed error $\mathcal{E}(\mathcal{S}; \boldsymbol{A}_{\mathcal{S}})$;
Verify $\mathcal{M}_l^{\mathrm{ser}}(\varnothing) \le \mathcal{B}_l$ and all initial expert-record capacities;
while *unselected candidates remain and the evaluation cap is not reached* **do**
  Form all unselected candidates fitting both the serialized layer budget and their expert-record capacity;
  if *no candidate is feasible* **then**
    stop;
  Refresh their fixed-factor screening scores and order them into bounded batches;
  accepted $\leftarrow$ false;
  for *each ordered candidate batch, until acceptance or the cap* **do**
    for *each candidate a in the batch* **do**
      Compute $\boldsymbol{Z}_{\mathcal{S}\cup\{a\}}$ through the current nonlinear expert;
      Solve Eq. (31) with fixed reference $\boldsymbol{A}_0$;
      Requantize the solution and compare it with $\boldsymbol{A}_{\mathcal{S}}$;
      Choose $\boldsymbol{A}_a$ by Eq. (32); divide its exact gain by the occupied-payload increment;
    if *an evaluated candidate has gain above $\varepsilon_{\mathrm{acc}}$* **then**
      Choose the highest feasible utility among candidates with gain above $\varepsilon_{\mathrm{acc}}$;
      Commit $\mathcal{S} \leftarrow \mathcal{S} \cup \{a\}$ and $\boldsymbol{A}_{\mathcal{S}} \leftarrow \boldsymbol{A}_a$;
      Update latent/output caches, error, occupied storage, and padding; mark accepted and leave the batch loop;
  if accepted $=$ false **then**
    stop; record exhaustion or evaluation-cap termination;
Serialize selected residuals, required scales/metadata, the final shared factor, and any fixed-record padding;
Return $(\mathcal{S}, \boldsymbol{A}_{\mathcal{S}})$;

No candidate sign or scale is refitted inside the selection loop. Expanding batches when none improves the state prevents a negative preliminary score alone from being mistaken for an impossibility result. The method can stop with unused budget and does not promise the best subset of the same cost.

A.3.6 BINARY FITTING PRIMITIVE AND RETAINED SCALES

For a target matrix $\boldsymbol{F} \in \mathbb{R}^{r\times I}$ and fixed signs $\boldsymbol{B}$, an alternating row/column fit minimizes

$$\|\boldsymbol{F} - \mathrm{diag}(\boldsymbol{u})\boldsymbol{B}\,\mathrm{diag}(\boldsymbol{v})\|_F^2 + \epsilon_{\mathrm{fit}}(\|\boldsymbol{u}\|_2^2 + \|\boldsymbol{v}\|_2^2), \qquad \epsilon_{\mathrm{fit}} > 0. \tag{62}$$

Holding the opposite scale vector fixed gives the coordinate minimizers

$$u_i = \frac{\sum_j F_{ij}B_{ij}v_j}{\sum_j (B_{ij}v_j)^2 + \epsilon_{\mathrm{fit}}}, \qquad v_j = \frac{\sum_i F_{ij}B_{ij}u_i}{\sum_i (B_{ij}u_i)^2 + \epsilon_{\mathrm{fit}}}. \tag{63}$$

The stabilizing denominator is explicitly associated with this regularizer, not described as an exact unregularized solution. This is a binary reconstruction primitive of the established row/column-scaling family (Li et al., 2024), not a new saliency mechanism. The same binary fitter and retained scale format are used across controlled comparisons.

Algorithm 6: One-Plane Binary Fit Used for Initialization and Candidate Construction

Input: Target $\boldsymbol{F}$; iteration limit; $\epsilon_{\mathrm{fit}} > 0$; convergence tolerance; retained scale rounding $\mathcal{Q}_s$
Output: Signs $\boldsymbol{B}$ and stored scales $\boldsymbol{u}, \boldsymbol{v}$

$\boldsymbol{B} \leftarrow \mathrm{sign}(\boldsymbol{F})$, with zero mapped to $+1$;
$\boldsymbol{u} \leftarrow \mathbf{1}_r$, $\boldsymbol{v} \leftarrow \mathbf{1}_I$;
for *each allowed fitting iteration* **do**
    Update every row scale using Eq. (63) with current column scales;
    Update every column scale using Eq. (63) with updated row scales;
    if *relative scale change meets the prescribed tolerance* **then**
        stop;
$\boldsymbol{u} \leftarrow \mathcal{Q}_s(\boldsymbol{u})$, $\boldsymbol{v} \leftarrow \mathcal{Q}_s(\boldsymbol{v})$;
Return $\boldsymbol{B}, \boldsymbol{u}, \boldsymbol{v}$; do not store a dense full-precision copy of the fitted matrix;

This primitive holds signs fixed while fitting scales; RCBF's subsequent sign refinement uses the routed nonlinear objective. For CERA, the output signs and scales are frozen as candidate data.

A.3.7 Calibration Configuration and Numerical Safeguards

The model-specific ranks, stored formats, 16-column residual groups, and record capacities are specified in Appendix A.4.2. The default calibration set contains 128 WikiText-2 training sequences of 4,096 tokens, and the default MOBR temperature is $\tau = 1.0$; their sensitivity is examined in Appendices A.4.7 and A.4.8. The algorithms take positive damping parameters $\lambda_{\mathrm{R}}, \xi, \lambda, \rho, \epsilon_{\mathrm{fit}}$, step-size schedules, iteration/evaluation limits, shortlist widths, and loss-specific acceptance tolerances as explicit inputs. Candidate ordering and tie-breaking are deterministic, and retained values and losses are checked for finiteness before a proposal can be committed. Non-finite proposals are rejected rather than serialized. The evaluations called "exact" refer to the stated finite calibration objective, up to numerical precision, rather than the end-to-end language-modeling loss. Calibration-loss acceptance, held-out perplexity, and downstream task accuracy are therefore distinct quantities.

A.4 More Experimental Results

A.4.1 Calibration and Search Hyperparameters

Unless otherwise stated, we use the same optimization and search controls across all evaluated models. Table 6 summarizes the numerical settings used by MOBR, RCBF, CERA, and the binary fitting primitive. Model-specific factor ranks and storage configurations are reported separately in Appendix A.4.2. The default calibration set contains 128 randomly sampled WikiText2 sequences of length 4,096, and the residual column-group width is 16. All acceptance tests evaluate the deployed, quantized candidate rather than its continuous surrogate.

For RCBF projected updates, the initial step size is reduced geometrically by the backtracking factor until a deployed candidate satisfies the acceptance criterion or the maximum number of trials is reached. MOBR and RCBF stop early when a complete sweep accepts no update. For CERA, candidate utilities are refreshed after each accepted residual block; at each state, candidates are screened using the current fixed-factor score and evaluated in shortlists of 64. Selection terminates when no feasible candidate yields an improvement larger than $\varepsilon_{\mathrm{acc}}$, the per-layer evaluation cap is reached, or the storage budget is exhausted. The binary fitting primitive alternates row- and column-scale updates until either the relative scale change falls below the tolerance or the iteration limit is reached.

A.4.2 Model-Specific Active-Path and Whole-Model Bitwidth

We instantiate the accounting in Appendix A.2 using the official checkpoint configurations and the fixed-record deployment layout. The values below report effective bitwidth only; they are not used to estimate absolute model size, runtime memory, or memory traffic. The target set includes routed-expert projections, attention projections, routers, and any dense or always-active expert projections. Token embeddings, the untied language-model head, and all normalization weights remain in BF16. Shared RCBF factors use symmetric INT8 with one BF16 scale per output row, while first-plane factors and residuals use packed binary signs and BF16 row/column scales. The gate and up ranks

Table 6: Optimization and search hyperparameters used by BitMoE. Unless otherwise stated, the same values are used across models.

Module	Hyperparameter	Value
MOBR	Temperature τ	1.0
	Hessian damping ρ	1×10^{-4}
	Maximum optimization sweeps	3
	Sign proposals per column	64
	Acceptance tolerance	1×10^{-6}
RCBF	Down-factor damping λ_{R}	1×10^{-3}
	Initial projected step size	1×10^{-2}
	Backtracking factor	0.5
	Maximum backtracking trials	5
	Maximum refinement sweeps	3
	Sign proposals per factor	128
CERA	Residual-fit damping ξ	1×10^{-4}
	Down-factor readaptation damping λ	1×10^{-3}
	Residual group width	16
	Shortlist width	64
	Candidate-evaluation cap per layer	4096
	Acceptance tolerance $\varepsilon_{\mathrm{acc}}$	1×10^{-6}
Binary fitting	Scale regularization ϵ_{fit}	1×10^{-6}
	Maximum alternating iterations	20
	Relative convergence tolerance	1×10^{-5}

equal the expert intermediate width h, while the down rank is fixed to 1,792 for all three models. CERA uses 16-column groups and selects residuals under fixed expert-record capacities. Unused capacity is counted in the reported static and active bitwidths.

For each model, the expert-record capacity is chosen to satisfy the 1.11 static target-weight BPW budget under the accounting in Appendix A.2, including shared factors, retained scales, residuals, metadata, and padding. Let L_{M} be the number of MoE layers, $P_e = 3dh$ the original parameter count per routed expert, P_D the original always-used target parameters, and P_U the retained non-target parameters. Let S_q, S_{act}, and S_U denote the corresponding serialized storage in bits. We compute

$$\begin{aligned} P_q &= L_{\mathrm{M}} E P_e + P_D, \qquad P_{\mathrm{act}} = L_{\mathrm{M}} k P_e + P_D, \\ B_{\mathrm{static}} &= \frac{S_q}{P_q}, \qquad B_{\mathrm{active}} = \frac{S_{\mathrm{act}}}{P_{\mathrm{act}}}, \\ B_{\mathrm{whole}} &= \frac{S_q + S_U}{P_q + P_U}. \end{aligned} \tag{64}$$

Shared factors are counted once per MoE projection, while only k routed expert records enter the active-path storage. The excess over one static bit results jointly from factor dimensions, shared INT8 storage, scales, residual capacity, metadata, and padding.

OLMoE-1B-7B-0125. The official checkpoint has $L_{\mathrm{M}} = 16$, $d = 2{,}048$, $h = 1{,}024$, $E = 64$, $k = 8$, and vocabulary size $V = 50{,}304$ (Allen Institute for AI, 2025). All 16 attention heads have dimension 128, with 16 key/value heads, giving four $2{,}048 \times 2{,}048$ attention matrices per layer. The RCBF ranks are $(r_{\mathrm{gate}}, r_{\mathrm{up}}, r_{\mathrm{down}}) = (1{,}024, 1{,}024, 1{,}792)$. Using Eq. (64) gives

$$B_{\mathrm{static}} = 1.1099705 \simeq 1.11, \qquad B_{\mathrm{active}} = 1.6904544 \simeq 1.69, \qquad B_{\mathrm{whole}} = 1.5536673 \simeq 1.55. \tag{65}$$

The larger whole-model BPW reflects the retained embedding/head parameters, whereas the larger active-path BPW reflects less amortization of the shared factors across eight rather than 64 experts.

DeepSeek-V2-Lite. This checkpoint has 27 layers, of which the first is dense and the remaining $L_{\mathrm{M}} = 26$ are MoE layers. Its configuration specifies $d = 2{,}048$, $h = 1{,}408$, $E = 64$, $k = 6$, $V = 102{,}400$, and two always-active shared experts per MoE layer (DeepSeek-AI, 2024a). The shared branch is implemented as one MLP of width $2h = 2{,}816$, and the first dense MLP has width 10,944. For the MLA implementation, the query, compressed-KV, expanded-KV, and output matrices have shapes $3{,}072 \times 2{,}048$, $576 \times 2{,}048$, $4{,}096 \times 512$, and $2{,}048 \times 2{,}048$, respectively (DeepSeek-AI, 2024b). With RCBF ranks $(1{,}408, 1{,}408, 1{,}792)$, the resulting bitwidths are

$$B_{\mathrm{static}} = 1.1099949 \simeq 1.11, \qquad B_{\mathrm{active}} = 1.7187850 \simeq 1.72, \qquad B_{\mathrm{whole}} = 1.5077424 \simeq 1.51. \tag{66}$$

The active-path accounting includes all dense and shared-branch target weights but only six routed expert records per MoE layer; the two always-active shared experts are not counted as members of the routed Top-k set.

Qwen3-30B-A3B. The configuration specifies $L_{\mathrm{M}} = 48$, $d = 2{,}048$, $h = 768$, $E = 128$, $k = 8$, $V = 151{,}936$, 32 query heads, four key/value heads, and head dimension 128 (Qwen Team, 2025). Thus, the query/key/value/output shapes are $4{,}096 \times 2{,}048$, $512 \times 2{,}048$, $512 \times 2{,}048$, and $2{,}048 \times 4{,}096$, giving $P_{\mathrm{attn}} = 18{,}874{,}368$ parameters per layer. There is no always-active shared expert branch. With RCBF ranks $(768, 768, 1{,}792)$, the resulting bitwidths are

$$B_{\mathrm{static}} = 1.1099255 \simeq 1.11, \qquad B_{\mathrm{active}} = 1.7202441 \simeq 1.72, \qquad B_{\mathrm{whole}} = 1.4135305 \simeq 1.41. \tag{67}$$

The near-identical rounded static BPWs across the three models therefore do not imply identical active-path or whole-model BPWs.

A.4.3 Comparison with Joint-SVD-Based Expert Decomposition

Purpose and comparison scope. This study is designed to distinguish RCBF's binary-aware refinement from the shared decomposition used to initialize it. Sub-MoE (Li et al., 2025) performs cluster-wise joint decomposition followed by expert-factor merging; its extended variant also incorporates activation-aware truncation. We therefore include both unweighted and activation-weighted decomposition controls rather than treating ordinary SVD as the only alternative. To isolate representation refinement, the matched-structure comparison retains every expert and uses one shared basis per projection. These are explicitly adapted decomposition controls, not reproductions of the complete Sub-MoE merging pipeline. The question is whether optimizing the deployed binary representation provides benefits beyond initializing and binarizing shared factors.

Controlled configurations. The comparison covers OLMoE-1B-7B, DeepSeek-V2-Lite, and Qwen3-30B-A3B. All variants use the ranks, INT8 shared-factor format, binary reconstruction scales, and fixed expert-record capacities specified in Appendix A.4.2. They consequently have the same allocated weight-record storage within each model, rounding to 1.11 static target-weight BPW; this equality includes unused record padding. Calibration data, non-expert quantization, and the MOBR and CERA policies are held fixed. CERA is rerun for each representation, so its selected residual blocks need not coincide across variants. Shared-factor freezing in the controls below applies to the representation-refinement stage only; subsequent CERA down-factor readaptation is available to every variant.

A: Joint SVD + binary fitting. Apply layer-joint SVD, quantize the shared factors, and fit the expert-specific binary factors without aggregate-output refinement. This is the single-group, no-merging decomposition control. **B: Activation-weighted SVD + binary fitting.** Use input-covariance weighting and dewhitening for the same layer-joint decomposition, followed by the same binary fitting procedure. **C: Fixed-basis aggregate refinement.** Start from A and optimize binary codes and scales using Eq. (20), while freezing the shared factors during this stage. **D: Joint refinement without cross-expert error terms.** Update both shared factors and binary expert factors, but remove cross-expert residual inner products from the calibration objective. **E: Full RCBF.** Update the same variables using the complete routed aggregate-output objective. Controls C–E use the same calibration examples, sign-proposal limits, and objective-evaluation caps. These matched limits control the search budget but do not imply equal wall-clock time.

Table 7: Matched-structure comparison of decomposition and refinement controls. All configurations use the same fixed-record storage, rounding to 1.11 static target-weight BPW, and the same MOBR and CERA policies. PPL denotes WikiText-2 perplexity and Avg.7 denotes seven-task average zero-shot accuracy (%). Bold and underlining indicate the best and second-best results in each performance column, respectively.

		OLMoE-1B-7B		DeepSeek-V2-Lite		Qwen3-30B-A3B	
Configuration	**BPW**	**PPL↓**	**Avg.7↑**	**PPL↓**	**Avg.7↑**	**PPL↓**	**Avg.7↑**
A: Joint SVD + binary fit	1.11	16.83	47.62	18.42	40.51	14.95	54.84
B: Activation-weighted SVD	1.11	15.92	48.54	17.69	41.36	14.22	55.76
C: Fixed-basis refinement	1.11	14.87	49.89	16.72	42.85	13.38	57.61
D: Without cross-expert terms	1.11	<u>14.51</u>	<u>50.36</u>	<u>16.20</u>	<u>43.58</u>	<u>13.05</u>	<u>58.12</u>
E: Full RCBF	1.11	**13.76**	**51.41**	**15.31**	**44.63**	**12.26**	**59.85**

For a precise definition of D, let $\boldsymbol{\delta}_{t,e} = \widehat{f}_e(\widehat{\boldsymbol{x}}_t) - f_e^\star(\widehat{\boldsymbol{x}}_t)$, where both expert functions receive the same input. With routing weights fixed within a layer, the control objective is

$$\mathcal{L}_{\text{sep}} = \mathcal{L}_{\text{R}} - \frac{2}{T}\sum_t \sum_{e<f} \bar{p}_{t,e}\bar{p}_{t,f}\langle \boldsymbol{\delta}_{t,e}, \boldsymbol{\delta}_{t,f}\rangle. \tag{68}$$

The teacher target, input drift, and linear residual terms are retained; only the cross-expert error terms are removed. Both candidate optimization and acceptance in D use this surrogate. Thus, its difference from E is not caused by replacing the teacher target or silently retaining aggregate-error acceptance.

Evaluation protocol. The comparison uses 128 WikiText-2 training sequences of length 4,096. A separate set of 32 validation sequences is reserved for reconstruction diagnostics and is not used to fit factors or select residuals. For paired layer-level diagnostics, all variants receive the same cached teacher targets, quantized-prefix inputs, and deployed routing weights. Before CERA, we measure the normalized aggregate reconstruction error

$$\text{NRE} = \frac{1}{L_{\text{M}}}\sum_{l=1}^{L_{\text{M}}} \frac{\sum_t \|\boldsymbol{y}_{l,t}^\star - \widehat{\boldsymbol{y}}_{l,t}\|_2^2}{\sum_t \|\boldsymbol{y}_{l,t}^\star\|_2^2 + \epsilon}, \qquad \epsilon = 10^{-12}, \tag{69}$$

where the sums use the held-out diagnostic tokens and include the complete MoE output. This pre-CERA measurement separates base-representation quality from subsequent residual selection. After running the same CERA policy, the complete models are evaluated by WikiText-2 test perplexity and seven-task average zero-shot accuracy using their own deployed routing. Table 7 reports the resulting downstream perplexity and average zero-shot accuracy.

Results and interpretation. Activation-weighted initialization (B) improves over unweighted joint SVD (A), but remains below the refinement variants in Table 7. The comparison between C and E shows the benefit of adapting shared factors jointly with binary expert factors rather than retaining a fixed basis. For example, on Qwen3-30B-A3B, E reduces perplexity from 13.38 to 12.26 and increases average accuracy from 57.61% to 59.85% relative to C. The comparison between D and E isolates cross-expert error coupling with the same teacher targets and update variables: retaining these terms reduces perplexity from 13.05 to 12.26 and improves average accuracy from 58.12% to 59.85% on the same model. Consistent improvements on OLMoE and DeepSeek-V2-Lite support the value of binary-aware joint refinement beyond decomposition alone. These conclusions concern the matched-structure controls, not the complete cluster-and-merge Sub-MoE pipeline, which changes expert counts and the number of shared bases.

A.4.4 Shared-Factor Precision and Bitwidth Analysis

We further analyze the effect of the precision used to store the shared RCBF factors $\boldsymbol{A}_m$. Experiments are conducted on OLMoE-1B-7B-0125 while keeping the RCBF ranks, binary expert factors, CERA storage capacities, scale formats, and calibration settings unchanged. For each setting, we

Table 8: Effect of shared-factor precision on OLMoE-1B-7B-0125. Static BPW includes shared factors, binary payloads, retained scales, CERA record capacity, and metadata. Bold and underlining mark the best and second-best compressed results per column; BF16 is the reference.

Shared Precision	Static BPW↓	Wiki PPL↓	Avg.7↑
BF16 Model	16.00	6.65	69.88
16-bit	1.22	<u>13.91</u>	<u>51.26</u>
8-bit	<u>1.11</u>	**13.76**	**51.41**
4-bit	**1.05**	15.48	49.07

re-quantize the shared factors at the specified precision and report static target-weight BPW under the accounting in Appendix A.4.2. With the BF16 row scales and descriptors unchanged, the shared-factor record at b bits is

$$\begin{aligned} A_b(o,r) &= 16 + a\left(\left\lceil \frac{bor}{8} \right\rceil\right) + a(2o), \\ S_A^{(b)} &= L_{\mathrm{M}}\left[2A_b(h,h) + A_b(d,1{,}792)\right], \\ B_{\mathrm{static}}^{(b)} &= \frac{8\left[S_q - S_A^{(8)} + S_A^{(b)}\right]}{P_q}. \end{aligned} \tag{70}$$

Here S_q is the 8-bit shared-factor baseline and all expert-record capacities remain fixed. For $b = 16, 8, 4$, this gives $B_{\mathrm{static}}^{(b)} = 1.2199362$, 1.1099705, and 1.0549877, respectively.

As shown in Table 8, the shared-factor precision has a clear impact on both storage and downstream performance. Using 16-bit shared factors yields 13.91 perplexity and 51.26% average zero-shot accuracy at 1.22 static BPW. Reducing the shared factors to 8-bit further lowers the storage cost to 1.11 BPW while slightly improving performance to 13.76 perplexity and 51.41% average accuracy. In this comparison, the 8-bit representation achieves slightly better task scores than the 16-bit representation while requiring less storage. Reducing the shared factors further to 4-bit lowers static BPW to 1.05, but increases perplexity to 15.48 and decreases average accuracy to 49.07%. Therefore, the 8-bit setting provides the best accuracy–storage trade-off among the evaluated configurations and is used as the default throughout our experiments.

A.4.5 Comparison of Residual Allocation Strategies

Comparison Setup. We further investigate whether CERA's improvement arises merely from adding a second binary residual plane or from its conditional allocation strategy. All variants start from the same MOBR-calibrated routing and RCBF representation, construct the same second-plane residual candidates, and use identical 16-column groups and per-expert storage capacities. The total serialized storage is therefore fixed to the same 1.11 static target-weight BPW for each model. The variants differ in their residual-selection criterion, score-refresh policy, and whether the shared down factor is readapted during allocation. We compare the following six allocation strategies: (i) *Random*, which uniformly samples feasible residual groups; (ii) *Magnitude*, which ranks candidates by their residual Frobenius norm per storage cost; (iii) *Static Saliency*, which applies the original GLAS-style Hessian-weighted saliency once before allocation; (iv) *One-shot Gain*, which ranks candidates by their exact output-reconstruction gain evaluated only at the empty selected set; (v) *Conditional Gain*, which recomputes reconstruction gains after every accepted candidate while keeping the shared factor fixed; and (vi) *Full CERA*, which additionally allows the shared down factor to readapt before evaluating each conditional gain.

For a candidate residual group a and the currently selected set $\mathcal{S}$, we define its storage-normalized conditional gain as

$$u(a \mid \mathcal{S}) = \frac{\mathcal{E}(\mathcal{S}; \boldsymbol{A}_{\mathcal{S}}) - \mathcal{E}(\mathcal{S} \cup \{a\}; \boldsymbol{A}_a)}{\mathcal{M}(\mathcal{S} \cup \{a\}) - \mathcal{M}(\mathcal{S})}, \tag{71}$$

where $\mathcal{E}$ denotes the routed reconstruction error and $\mathcal{M}$ is the occupied packed storage, excluding unused record capacity. One-shot Gain evaluates the fixed-factor gain per occupied bit only at $\mathcal{S} = \varnothing$ and keeps that ranking fixed throughout allocation. Conditional Gain refreshes the ranking after

Table 9: Comparison of residual allocation strategies under the same 1.11 static target-weight BPW. All methods share MOBR and RCBF configurations, residual candidates, and record capacities. Bold and underlining indicate the best and second-best performance per column.

	OLMoE-1B-7B		DeepSeek-V2-Lite		Qwen3-30B-A3B	
Allocation Strategy	**PPL↓**	**Avg.7↑**	**PPL↓**	**Avg.7↑**	**PPL↓**	**Avg.7↑**
Random	15.62	48.62	17.48	41.80	14.07	56.32
Magnitude	15.08	49.21	16.95	42.34	13.64	57.01
Static Saliency	14.72	49.77	16.57	42.90	13.29	57.49
One-shot Gain	14.41	50.13	16.23	43.31	12.94	58.05
Conditional Gain	<u>14.10</u>	<u>50.69</u>	<u>15.83</u>	<u>43.96</u>	<u>12.61</u>	<u>58.74</u>
Full CERA	**13.76**	**51.41**	**15.31**	**44.63**	**12.26**	**59.85**

every selection but retains $\boldsymbol{A}_a = \boldsymbol{A}_{\mathcal{S}} = \boldsymbol{A}_0$. Full CERA uses Eq. (32) to compare the unchanged factor against the rounded readaptation proposal, and retains the lower-error option. Every variant obeys the same layer budget and per-expert record capacities. The fixed-score controls follow their stated sampling or ranking rules under these capacities. Conditional Gain and Full CERA stop when no feasible candidate has a current gain above the acceptance tolerance, no candidate fits, or the evaluation cap is reached. Unused capacity remains serialized as padding, so early stopping does not change the allocated static BPW.

Results. As shown in Table 9, random residual placement performs consistently worse than all structure-aware allocation strategies, showing that selecting informative residual groups matters even when per-expert record capacities are fixed. Magnitude-based allocation provides a clear improvement, but remains inferior to Static Saliency, indicating that residual amplitude alone does not reliably reflect its downstream reconstruction utility. Replacing static saliency with the exact One-shot Gain further improves all three models, showing that evaluating candidates in the routed output space better captures their actual contribution. However, keeping this ranking fixed remains suboptimal because the benefit of one residual depends on previously selected corrections. Refreshing the candidate utilities through Conditional Gain therefore yields additional improvements, reducing WikiText-2 perplexity from 12.94 to 12.61 and increasing average zero-shot accuracy from 58.05% to 58.74% on Qwen3-30B-A3B. Finally, allowing the shared factor to readapt during candidate evaluation further improves performance to 12.26 perplexity and 59.85% average accuracy. Similar trends hold on OLMoE and DeepSeek-V2-Lite, demonstrating that both conditional score refresh and shared-factor readaptation contribute to CERA rather than merely increasing residual capacity.

A.4.6 Ablation Study on Routing Similarity

We analyze routing-probability similarity and Top-k expert consistency on Qwen1.5-MoE-A2.7B to examine post-quantization *expert-shift*. The former measures agreement between routing distributions; the latter measures overlap between the selected expert sets.

Routing Probability Similarity. Given router logits $\mathbf{z} \in \mathbb{R}^E$ for E experts, we first obtain the routing probability vector

$$\mathbf{p} = \mathrm{softmax}(\mathbf{z}) \in \Delta^{E-1}, \tag{72}$$

where Δ^{E-1} denotes the $(E-1)$-dimensional probability simplex. For token t, let $\mathbf{p}_{\mathrm{FP}}^{(t)}$ and $\mathbf{p}_{\mathrm{Q}}^{(t)}$ denote the routing probabilities of the full-precision and quantized models, respectively. We measure their similarity using

$$s^{(t)} = \frac{\left\langle \mathbf{p}_{\mathrm{FP}}^{(t)}, \mathbf{p}_{\mathrm{Q}}^{(t)} \right\rangle}{\left\| \mathbf{p}_{\mathrm{FP}}^{(t)} \right\|_2 \left\| \mathbf{p}_{\mathrm{Q}}^{(t)} \right\|_2}. \tag{73}$$

The layer-wise routing similarity is then

$$S_\ell = \frac{1}{T} \sum_{t=1}^{T} s^{(t)}, \tag{74}$$

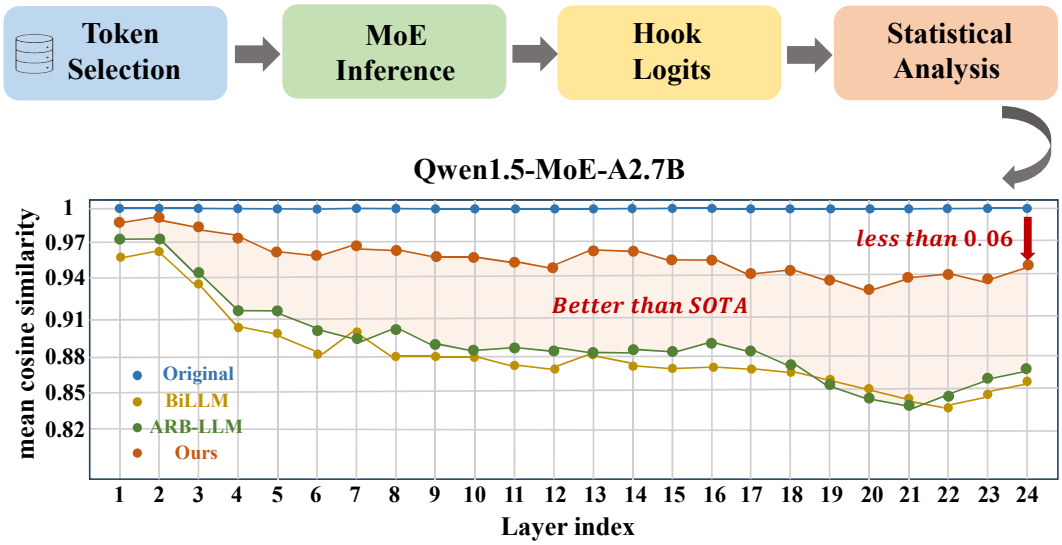


Figure 7: Layer-wise routing-probability similarity between full-precision and binary models on Qwen1.5-MoE-A2.7B.

where T is the number of evaluated tokens.

Top-k Expert Consistency. High probability similarity does not necessarily imply identical expert selection. We therefore additionally measure the overlap between the full-precision and quantized Top-k expert sets. Let $\mathcal{S}_{\mathrm{FP}}^{(t)}$ and $\mathcal{S}_{\mathrm{Q}}^{(t)}$ denote the corresponding selected experts. The layer-wise Top-k consistency is defined as

$$C_\ell = \frac{1}{T}\sum_{t=1}^{T}\frac{\left|\mathcal{S}_{\mathrm{FP}}^{(t)} \cap \mathcal{S}_{\mathrm{Q}}^{(t)}\right|}{k}. \tag{75}$$

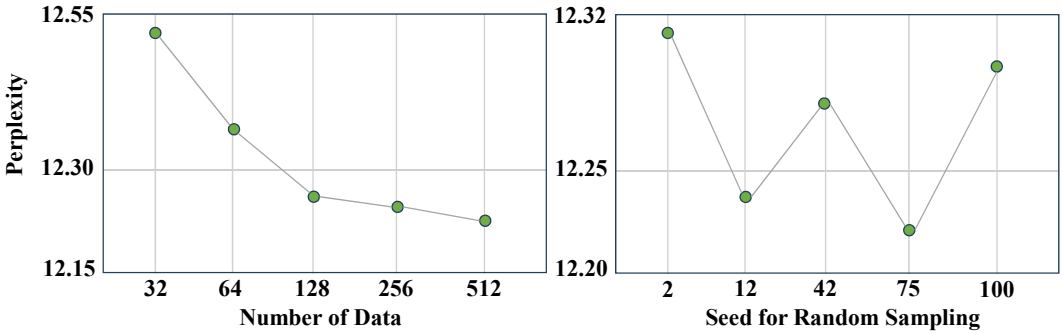
This metric measures the average fraction of selected experts shared with the full-precision model; it is not the fraction of tokens with an entirely identical Top-k set.

Results and Analysis. On WikiText-2, we compare the routing of BiLLM, ARB-LLM, and BitMoE against the full-precision model. As shown in Fig. 7, BiLLM and ARB-LLM exhibit progressively lower routing-probability similarity in deeper layers, with several layers falling below 0.85. In contrast, BitMoE with MOBR maintains $S_\ell \geq 0.9$ throughout the network, indicating substantially closer agreement with the full-precision routing distribution. The Top-k consistency analysis shows the same trend: averaged across layers, BiLLM and ARB-LLM preserve approximately 80.6% and 84.3% of the full-precision expert selections, respectively, whereas BitMoE increases the overlap to 93.7%. These results show that BitMoE preserves both routing-probability similarity and selected-expert overlap more effectively than the compared binary backbones, consistent with the routing-preservation objective of MOBR.

A.4.7 Ablation Study on Calibration Data

We further study the sensitivity of the complete BitMoE calibration pipeline to calibration-set size and random sampling. All experiments are conducted on Qwen3-30B-A3B using WikiText-2. For every setting, we rerun the complete layerwise calibration procedure, including MOBR, RCBF, and CERA, while keeping all other hyperparameters and the 1.11 static target-weight BPW budget fixed. We then evaluate perplexity on the same held-out WikiText-2 split.

Calibration Set Size. We evaluate $\{32, 64, 128, 256, 512\}$ calibration sequences, each containing 4,096 tokens. As shown in Fig. 8, increasing the calibration size from 32 to 128 steadily improves perplexity from 12.54 to 12.26. Further increasing the calibration set to 256 and 512 sequences



(a) Change of Number **(b) Change of Seed**

Figure 8: Sensitivity of BitMoE to calibration data on Qwen3-30B-A3B. Left: WikiText-2 perplexity under different numbers of calibration sequences. Right: perplexity across five random calibration seeds with 128 sequences.

yields only marginal improvements to 12.21 and 12.19, respectively. This trend indicates that the calibration process benefits from additional samples at small data scales but largely saturates around 128 sequences, which we therefore use as the default setting.

Random Seed Sensitivity. We additionally fix the calibration size to 128 sequences and vary the random seed controlling both sequence sampling and ordering. Across five seeds, the resulting WikiText-2 perplexities are $\{12.31, 12.24, 12.27, 12.22, 12.29\}$, corresponding to 12.27 ± 0.04 (mean $\pm$ sample standard deviation). The small variation indicates stable performance of the complete MOBR–RCBF–CERA pipeline across these calibration samples.

A.4.8 Ablation Study on the Temperature Parameter τ

We further investigate the sensitivity of Margin-Aware Optimal Brain Routing (MOBR) to the temperature parameter τ. In MOBR, τ controls the sharpness of the pairwise routing probabilities, $q_{ij}^{\star} = \sigma(m_{ij}^{\star}/\tau)$ and $q_{ij}^{(0)} = \sigma(m_{ij}^{(0)}/\tau)$, and consequently affects both the first- and second-order terms used for local router reconstruction. Specifically, the gradient contribution scales with $(q_{ij}^{(0)} - q_{ij}^{\star})/\tau$, while the curvature contribution scales with $q_{ij}^{(0)}(1 - q_{ij}^{(0)})/\tau^2$. Thus, τ controls the softness of the teacher targets and the concentration of curvature around the current student boundary, while the gradient also reflects teacher–student disagreement.

We evaluate $\tau \in \{0.5, 1.0, \ldots, 4.5\}$ on DeepSeek-V2-Lite and Qwen3-30B-A3B while keeping all other quantization settings fixed. As shown in Fig. 9, WikiText-2 perplexity exhibits a U-shaped trend on both models.

- **Over-sharpening at small τ.** When τ is too small, the pairwise sigmoid becomes highly saturated for all but a narrow set of near-boundary pairs. The curvature weights concentrate around nearly zero student margins. However, a confidently reversed teacher–student pair can still produce a large gradient despite having almost zero curvature. This mismatch can make the local quadratic update poorly conditioned and overly dependent on damping.
- **Loss of boundary selectivity at large τ.** As τ increases, the pairwise probabilities move toward 0.5 and their differences are compressed. For fixed finite margins, the gradient approaches $(m_{ij}^{(0)} - m_{ij}^{\star})/(4\tau^2)$ and the curvature approaches $1/(4\tau^2)$. The curvature therefore becomes less selective across pairs, and both derivative terms weaken relative to fixed damping. These properties provide possible explanations for the observed sensitivity, rather than a guarantee of the downstream perplexity trend.

Across both models, $\tau = 1.0$ gives the lowest perplexity, while settings in the range $\tau \in [1.0, 2.0]$ remain close to the optimum. We therefore use $\tau = 1.0$ as the default throughout our experiments. The relatively flat region around the default also indicates that MOBR is not highly sensitive to moderate variations of τ.

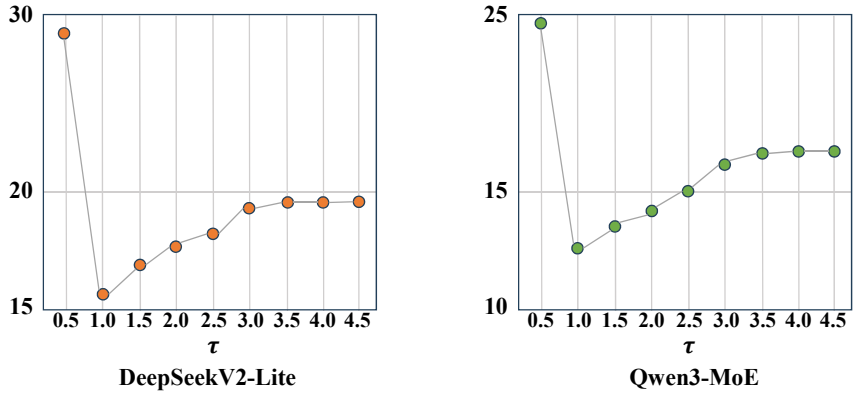


Figure 9: Ablation of the MOBR temperature τ on DeepSeek-V2-Lite and Qwen3-30B-A3B using WikiText-2 perplexity. Small τ concentrates curvature near zero student margins, whereas large τ weakens curvature selectivity. $\tau = 1.0$ achieves the best performance on both models, with relatively stable results in the range $[1.0, 2.0]$.

A.4.9 Plug-and-Play Ablations of MOBR

To evaluate the plug-and-play applicability of MOBR, we integrate it into two representative binary PTQ backbones, BiLLM and ARB-LLM, without modifying their original weight-quantization procedures. We evaluate WikiText-2, C4, and PTB perplexity on OLMoE-1B-7B and DeepSeek-V2-Lite. For each backbone, we compare the original quantized model against its MOBR-enhanced counterpart while keeping the calibration data, tokenization, weight-quantization settings, and inference hyperparameters unchanged. No additional full-model fine-tuning is involved beyond router calibration.

As shown in Fig. 10 and Fig. 11, MOBR consistently reduces perplexity across all twelve backbone–model–dataset combinations. For BiLLM, MOBR yields an average relative perplexity reduction of approximately 8.6%, decreasing WikiText-2 perplexity from 20.32 to 18.61 on OLMoE-1B-7B and from 22.43 to 20.31 on DeepSeek-V2-Lite. For ARB-LLM, the average reduction is approximately 6.4%, with the largest improvement observed on DeepSeek-V2-Lite PTB, where perplexity decreases from 52.74 to 46.69. Overall, the relative reductions range from 3.6% to 11.5%, demonstrating that explicitly preserving routing decisions provides complementary benefits beyond the underlying binary weight reconstruction.

These results support the plug-and-play compatibility of MOBR with different binary PTQ backbones. Since MOBR operates only on router calibration, these improvements are obtained without changing the backbone-specific expert-weight quantizer or introducing full-model retraining, further indicating that quantization-induced expert-shift constitutes a complementary source of degradation in MoE binarization.

A.5 Dialog Examples

To qualitatively examine how extreme binarization affects generation, Table 10 presents representative responses from OLMoE-1B-7B, DeepSeek-V2-Lite, and Qwen3-30B-A3B in full precision and after compression with BiLLM, ARB-LLM, and BitMoE. In the displayed examples, BiLLM and ARB-LLM fail to provide the requested information, with some outputs degenerating into repetitions, fragmented text, or irrelevant content. In contrast, BitMoE largely preserves response fluency and semantic relevance, producing complete answers that retain the essential information conveyed by the full-precision models. These qualitative examples are consistent with the quantitative results, further illustrating BitMoE's ability to preserve generative capability under extreme compression.

A.6 Limitations

BitMoE primarily targets weight binarization rather than joint weight–activation quantization; expert activations remain in higher precision. Although the evaluation includes models up to 80B

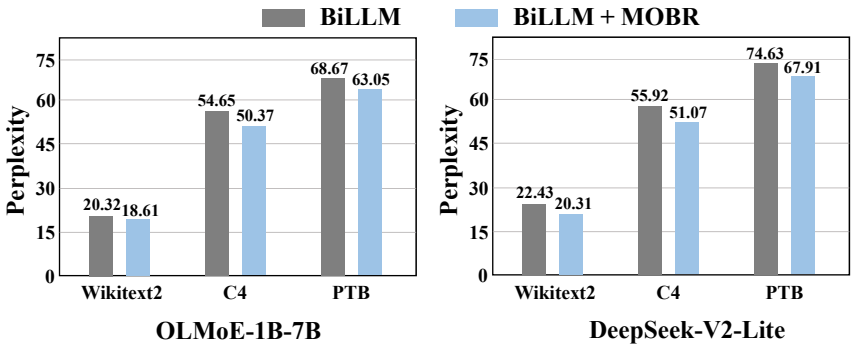


Figure 10: Plug-and-play evaluation of MOBR on the BiLLM backbone. Perplexity on WikiText-2, C4, and PTB for OLMoE-1B-7B and DeepSeek-V2-Lite. Bars compare the original BiLLM model with BiLLM + MOBR under identical weight-quantization settings.

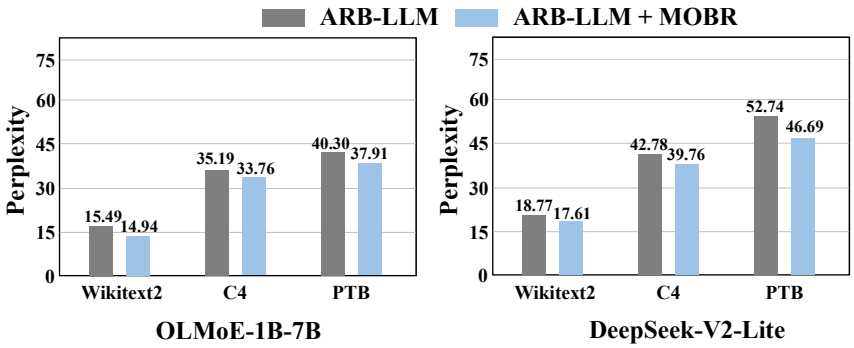


Figure 11: Plug-and-play evaluation of MOBR on the ARB-LLM backbone. Perplexity on WikiText-2, C4, and PTB for OLMoE-1B-7B and DeepSeek-V2-Lite. Bars compare the original ARB-LLM model with ARB-LLM + MOBR while keeping the underlying weight quantization unchanged.

Table 10: Representative generations from full-precision and binary MoE models. The **input** is shown in bold. Generated text is colored to indicate appropriate and inappropriate responses.

	OLMoE-1B-7B-0125
Original	**What are the capitals of the United States, the United Kingdom, and France respectively?** Washington D.C., London, Paris.
BiLLM	**What are the capitals of the United States, the United Kingdom, and France respectively?** The capitals of the United States, the United Kingdom, and France.
ARB-LLM	**What are the capitals of the United States, the United Kingdom, and France respectively?** The capitals of these three countries.
BitMoE	**What are the capitals of the United States, the United Kingdom, and France respectively?** They are Washington, D.C., London, and Paris, respectively.
	DeepSeek-V2-Lite
Original	**Who were the first three presidents of the United States?** The first three U.S. presidents were George Washington, John Adams and Thomas Jefferson.
BiLLM	**Who were the first three presidents of the United States?** The... Ss, s, s, s, s, s, s, s, s, s, s, s, s, s, s, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o, o). t, s, s, s, s, o, o ...
ARB-LLM	**Who were the first three presidents of the United States?** 50 Years Old And Now Have Been A Serious Moment.
BitMoE	**Who were the first three presidents of the United States?** The first three were George Washington, John Adams, and Thomas Jefferson, in that order.
	Qwen3-30B-A3B
Original	**How should I adjust if I didn't perform well in my final exams? Please give me a short answer.** If you didn't do well in your exams, reflect on what went wrong, seek help, and revise strategically. Prepare by starting early, using active learning, and managing your time. Improve memory through practice, good habits, and mnemonics. Effective techniques include spaced repetition, active recall, and interleaving. Stay focused and keep going—you can improve!
BiLLM	**How should I adjust if I didn't perform well in my final exams? Please give me a short answer.** Let me explain the situation. So, I want to explain the situation. Let me explain the situation. So, I want to explain the situation ...
ARB-LLM	**How should I adjust if I didn't perform well in my final exams? Please give me a short answer.** (The answer is 10-20 seconds) [please] [give me] [the answer] Please [please] ...
BitMoE	**How should I adjust if I didn't perform well in my final exams? Please give me a short answer.** Review where you lost points, adjust your study plan, and practice your weak areas consistently. Ask your teachers for targeted feedback when needed.

parameters, its behavior on trillion-parameter models remains unexplored. Calibration also depends on the coverage of the sampled routing patterns, and monotonic reconstruction-loss reduction does not guarantee improvements on unseen tasks. Finally, encoded weight savings do not directly specify peak runtime memory or speedup, which depend on batching, expert dispatch, dequantization, and kernel support.

A.7 Broader Impacts

BitMoE reduces target-weight resident storage by over 90% relative to BF16 and achieves a reported $2.34\times$ inference speedup, which can lower the resource barrier to deploying MoE models. These efficiency gains may enable broader access on resource-constrained hardware, but they do not by themselves quantify reductions in energy consumption or carbon emissions. Lower deployment costs may also facilitate misuse of LLMs, underscoring the importance of safety evaluation and appropriate deployment safeguards.