BOLD-MoE: Base-Optimized Learning of Orthogonal Dictionaries for Mixture-of-Experts Compression

Anonymous authors
Paper under double-blind review

Abstract

Recent Mixture-of-Experts (MoE) compression methods exploit inter-expert redundancy by representing each expert as a shared base plus a compressed expert-specific residual. However, the base is typically constructed before residual compression, ignoring that its optimal choice depends on how well the resulting residuals can be represented under a limited budget. We formulate shared-base MoE compression as a coupled optimization problem and solve it by alternating between residual compression and an activation-aware base update. This formulation is compressor-agnostic: even when retaining SVD residuals, alternating base–residual optimization improves over the one-shot decomposition. We further introduce orthogonal sparse dictionary residuals, which replace the single low-rank subspace imposed by SVD with a richer union-of-subspaces representation. Across three MoE architectures and 20–60% compression, BOLD-MoE consistently improves perplexity and zero-shot accuracy over the strongest shared-base baseline, with substantial gains persisting under strong compression.

1 Introduction

Mixture-of-Experts (MoE) models increase model capacity while activating only a small subset of experts per token (Fedus et al., 2022; Jiang et al., 2024; Dai et al., 2024; Yang et al., 2024a; 2025). Their expert parameters, however, still dominate storage, making compression important for deployment. A growing line of work exploits redundancy across experts by representing them through a shared base together with compact expert-specific residuals (Yang et al., 2024b; Li et al., 2025; Gu et al., 2025; Huang et al., 2025; Ai et al., 2025).

Several recent methods compress MoE experts by separating a shared component from lightweight expert-specific residuals (Huang et al., 2025; Gu et al., 2025; Ai et al., 2025). In these approaches, the shared component is constructed first and the induced expert-specific residuals are compressed afterwards. This sequential design ignores the coupling between the two: the residuals depend on the chosen base, while the best base depends on what the residual compressor can preserve under a limited budget. We therefore treat base construction and residual compression as a coupled problem and optimize them by block-coordinate descent, repeatedly updating one while fixing the other. This idea is independent of the residual compressor and improves performance even when retaining the same SVD residual representation.

The choice of residual geometry is equally important. MoE experts are explicitly encouraged to specialize, processing distinct token and semantic regions and capturing complementary knowledge (Dai et al., 2024; Nielsen et al., 2025; Li et al., 2026). A single shared base necessarily captures only the common structure across these experts, leaving the residuals to represent their heterogeneous, expert-specific deviations. Low-rank factorization is a natural and well-established choice for such compression (Chen et al., 2021; Hsu et al., 2022; Yuan et al., 2023; Wang et al., 2025; Gu et al., 2025), but rank-r SVD constrains every column of a residual matrix to the same r-dimensional subspace. This can be overly restrictive when different output directions require different expert-specific corrections. Motivated by recent dictionary-based LLM compression (Makhov et al., 2026a;b), we instead represent residuals with orthogonal sparse dictionaries, allowing each column to select its own subset of atoms and thereby model a union of subspaces.

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.

Expert specialization also complicates model-wide budget allocation. In dense models, dynamic allocation redistributes capacity across layers and projection types according to their estimated compression sensitivity. For MoEs, the same global competition additionally spans experts whose residuals may differ substantially in spectrum, activation statistics, and approximation difficulty. It is therefore unclear whether a dense-model allocation rule can compare these heterogeneous residuals directly, or whether the budget should be balanced within expert- or layer-specific groups. Rather than introducing a new allocation rule, we apply the one-shot global spectral allocator of COMPOT (Makhov et al., 2026b) to expert residuals and study whether its global pool should be restricted by MoE structure. Empirically, global allocation remains competitive with these grouped variants.

We introduce **BOLD-MoE** (**B**ase-**O**ptimized **L**earning of Orthogonal **D**ictionaries). Our contributions are threefold. *(i)* We formulate shared-base MoE compression as a coupled base–residual optimization problem and solve it by alternating updates, improving SVD-based compression without changing the residual family. *(ii)* We introduce orthogonal sparse dictionary residuals, replacing the single low-rank subspace of SVD with a more flexible union-of-subspaces representation. *(iii)* We study whether global budget allocation transfers from dense models to heterogeneous MoE residuals and find that it remains effective without MoE-specific grouping; across three MoE architectures and 20–60% compression, BOLD-MoE consistently improves perplexity and zero-shot accuracy over D^2-MoE.

2 Related Work

MoE compression and inter-expert sharing. MoE compression has been approached through expert pruning, intra-expert compression, and explicit sharing of parameters across experts. For example, MoE-I^2 (Yang et al., 2024b) combines inter-expert pruning with low-rank decomposition of the retained experts. A complementary line of work exploits redundancy between experts directly. MoE-SVD (Li et al., 2025) factorizes expert weights and shares one low-rank factor across experts, while MoBE (Chen et al., 2026) represents expert factors using combinations of basis matrices shared within each MoE layer. Sub-MoE (Li et al., 2026) first groups functionally similar experts and then merges them within a shared subspace. These approaches demonstrate that substantial expert structure can be shared without forcing all experts to become identical.

More closely related to BOLD-MoE are methods that explicitly decompose an expert into a common component and an expert-specific correction. DeRS (Huang et al., 2025) represents upcycled MoE experts as a shared base plus lightweight delta weights, while ResMoE (Ai et al., 2025) constructs a common expert through a Wasserstein barycenter and approximates the remaining residuals. D^2-MoE (Gu et al., 2025) uses Fisher-weighted expert merging to construct a shared base and subsequently compresses the induced delta weights with SVD. In these methods, the common component determines the residuals that are compressed afterwards. BOLD-MoE instead treats the shared base and compressed residuals as coupled variables: after residual compression, the base is revisited according to the remaining activation-aware approximation error.

Low-rank language-model compression. Low-rank factorization is a standard approach to neural-network compression, and several methods improve upon weight-space truncated SVD by incorporating data or curvature information. FWSVD (Hsu et al., 2022) introduces Fisher-weighted low-rank approximation, while ASVD (Yuan et al., 2023) and SVD-LLM (Wang et al., 2025) use activation statistics to obtain more function-aware decompositions. Such techniques naturally extend to MoE models, where both MoE-SVD and D^2-MoE use low-rank structure to reduce expert parameters (Li et al., 2025; Gu et al., 2025). However, a rank-r factorization represents all columns of a matrix through the same r-dimensional subspace. For heterogeneous expert residuals, this shared-subspace constraint can limit the directions available to represent expert-specific corrections.

Sparse dictionary learning and model-wide allocation. Sparse dictionary learning provides an alternative structured factorization in which each vector selects a sparse subset of atoms from a larger dictionary. Classical methods such as K-SVD (Aharon et al., 2006) jointly optimize the dictionary and sparse coefficients, yielding a union-of-subspaces representation rather than a single low-rank subspace. More recently, CoSpaDi (Makhov et al., 2026a) applied sparse dictionary learning to post-training LLM compression, while COMPOT (Makhov et al., 2026b) introduced an orthogonal formulation with closed-form sparse-code and Procrustes updates together with one-

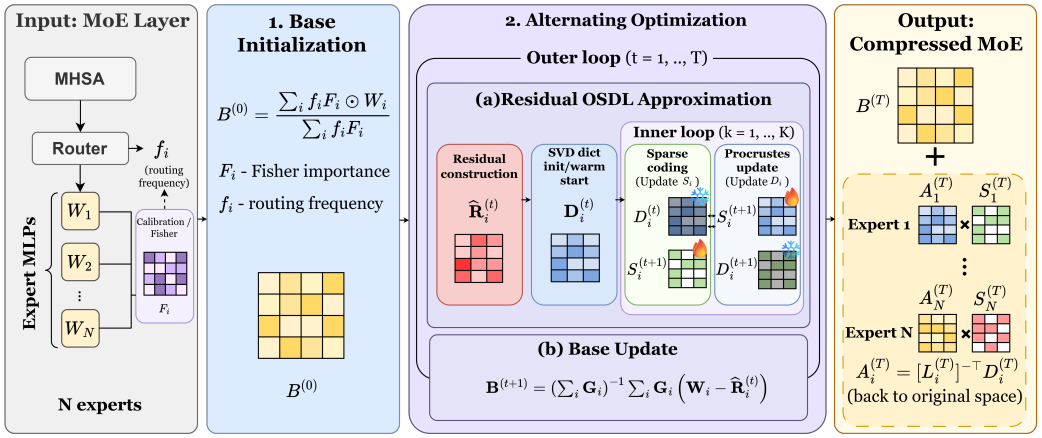


Figure 1: Overview of BOLD-MoE with orthogonal sparse residuals. Starting from an initial shared base, each outer iteration approximates the current expert residuals by OSDL and then updates the shared base according to the remaining activation-aware reconstruction error. The final experts are represented by the shared base and sparse expert-specific corrections.

shot model-wide budget allocation. BOLD-MoE adopts this orthogonal sparse representation for expert-specific residuals, but places it inside the coupled shared-base optimization problem. It also examines whether the global allocation strategy developed for dense models remains suitable when the allocation pool additionally spans experts with different routed activation distributions.

3 Method

BOLD-MoE starts from a simple observation: if an expert is represented by a shared base and a compressed residual, the two components should not be chosen independently. We first formulate this coupled problem, then solve it by block-coordinate descent (BCD). The formulation is agnostic to the residual compressor; we consider both low-rank SVD and orthogonal sparse dictionary learning (OSDL).

3.1 Coupled Shared-Base Compression

Consider one projection type in an MoE layer with N experts, with weights $\mathbf{W}_i \in \mathbb{R}^{d_{\text{in}} \times d_{\text{out}}}$. We represent each expert as

$$\mathbf{W}_i \approx \mathbf{B} + \widehat{\mathbf{R}}_i, \qquad \widehat{\mathbf{R}}_i \in \mathcal{C}_i, \tag{1}$$

where $\mathbf{B}$ is shared across experts and $\mathcal{C}_i$ denotes the admissible compressed residual family.

Let $\mathbf{X}_i \in \mathbb{R}^{n_i \times d_{\text{in}}}$ contain calibration activations routed to expert i, and define the raw activation Gram matrix

$$\mathbf{G}_i^{\text{raw}} = \mathbf{X}_i^\top \mathbf{X}_i. \tag{2}$$

Routed activations can yield rank-deficient Gram matrices, so when necessary we apply a small eigenvalue floor before Cholesky factorization and denote the resulting positive-definite matrix by

$$\mathbf{G}_i = \text{stab}(\mathbf{G}_i^{\text{raw}}) = \mathbf{L}_i \mathbf{L}_i^\top. \tag{3}$$

The stabilization procedure is specified in Appendix A.1.

Without stabilization, the reconstruction error on the calibration inputs is exactly

$$\left\| \mathbf{X}_i \left(\mathbf{W}_i - \mathbf{B} - \widehat{\mathbf{R}}_i \right) \right\|_F^2 = \text{tr}\left[\Delta_i^\top \mathbf{G}_i^{\text{raw}} \Delta_i \right], \qquad \Delta_i = \mathbf{W}_i - \mathbf{B} - \widehat{\mathbf{R}}_i. \tag{4}$$

We use its numerically stabilized counterpart and optimize

$$\min_{\mathbf{B}, \{\widehat{\mathbf{R}}_i\}} \sum_{i=1}^{N} \left\| \mathbf{L}_i^\top \left(\mathbf{W}_i - \mathbf{B} - \widehat{\mathbf{R}}_i \right) \right\|_F^2 \quad \text{s.t.} \quad \widehat{\mathbf{R}}_i \in \mathcal{C}_i \ \forall i. \tag{5}$$

3.2 Alternating Base–Residual Optimization

Joint optimization of equation 5 is generally difficult, but its two blocks are simple when considered separately. BOLD-MoE therefore alternates between compressing the residuals for a fixed base and updating the base for the current compressed residuals.

Residual update. Given $\mathbf{B}^{(t)}$, each expert is independent and solves

$$\widehat{\mathbf{R}}_i^{(t+1)} \approx \underset{\widehat{\mathbf{R}} \in \mathcal{C}_i}{\arg\min} \left\| \mathbf{L}_i^\top (\mathbf{W}_i - \mathbf{B}^{(t)} - \widehat{\mathbf{R}}) \right\|_F^2 . \tag{6}$$

Thus, equation 6 can use any compressor compatible with $\mathcal{C}_i$.

Base update. For fixed compressed residuals, equation 5 is quadratic in $\mathbf{B}$. Its exact minimizer satisfies

$$\left(\sum_{i=1}^{N} \mathbf{G}_i \right) \mathbf{B}^{(t+1)} = \sum_{i=1}^{N} \mathbf{G}_i \left(\mathbf{W}_i - \widehat{\mathbf{R}}_i^{(t+1)} \right), \tag{7}$$

and hence

$$\mathbf{B}^{(t+1)} = \left(\sum_{i=1}^{N} \mathbf{G}_i \right)^{-1} \sum_{i=1}^{N} \mathbf{G}_i \left(\mathbf{W}_i - \widehat{\mathbf{R}}_i^{(t+1)} \right). \tag{8}$$

The update has a useful interpretation: the shared base moves toward structure that the current compressed residuals fail to represent. The next residual update is then computed relative to this corrected base, and the process repeats.

The framework does not require a particular initialization. In our experiments, we initialize $\mathbf{B}^{(0)}$ with the Fisher- and routing-frequency-weighted merge of D^2-MoE (Gu et al., 2025), so iteration zero exactly matches the one-shot shared-base construction used by the baseline.

Each exact block update is non-increasing in equation 5. Since the objective is bounded below, its sequence of values converges. The overall problem remains non-convex for the residual families considered below, so this does not imply global optimality. We provide the corresponding monotonicity argument in Appendix A.1.

3.3 Residual Compression

For a fixed base, define the activation-whitened residual

$$\widetilde{\mathbf{R}}_i^{(t)} = \mathbf{L}_i^\top \left(\mathbf{W}_i - \mathbf{B}^{(t)} \right). \tag{9}$$

The residual block is therefore an ordinary Frobenius-norm approximation of $\widetilde{\mathbf{R}}_i^{(t)}$.

Low-rank residuals. A direct choice is truncated SVD. If

$$\widetilde{\mathbf{R}}_i = \mathbf{U}_i \mathbf{\Sigma}_i \mathbf{V}_i^\top, \tag{10}$$

a rank-r_i approximation gives

$$\widehat{\mathbf{R}}_{i,\mathrm{SVD}} = \mathbf{L}_i^{-\top} \mathbf{U}_{i,r_i} \mathbf{\Sigma}_{i,r_i} \mathbf{V}_{i,r_i}^\top. \tag{11}$$

This provides a direct SVD-based instance of our BCD formulation and isolates the effect of re-optimizing the shared base. A rank-r_i approximation constrains all columns of the residual to the same r_i-dimensional subspace.

Orthogonal sparse residuals. In contrast, sparse dictionary learning provides a union-of-subspaces representation: each residual column can select its own sparse subset of atoms from a dictionary with more available directions than the corresponding truncated SVD rank. Figure 1 summarizes BOLD-MoE with this OSDL residual parameterization. We factorize

$$\widetilde{\mathbf{R}}_i \approx \mathbf{D}_i \mathbf{S}_i, \tag{12}$$

where $\mathbf{D}_i \in \mathbb{R}^{d_{\text{in}} \times k_i}$, with $k_i \leq d_{\text{in}}$, is an orthogonal dictionary and $\mathbf{S}_i \in \mathbb{R}^{k_i \times d_{\text{out}}}$ contains sparse coefficients:

$$\min_{\mathbf{D}_i, \mathbf{S}_i} \left\| \widetilde{\mathbf{R}}_i - \mathbf{D}_i \mathbf{S}_i \right\|_F^2 \quad \text{s.t.} \quad \mathbf{D}_i^\top \mathbf{D}_i = \mathbf{I}, \qquad \|\mathbf{s}_{i,j}\|_0 \leq s_i. \tag{13}$$

Following COMPOT (Makhov et al., 2026b), orthogonality makes both conditional updates simple. We initialize $\mathbf{D}_i$ from the leading k_i left singular vectors of $\widetilde{\mathbf{R}}_i^{(0)}$ and warm-start it across subsequent outer iterations. For fixed $\mathbf{D}_i$, sparse coding is

$$\mathbf{S}_i \leftarrow \mathcal{H}_{s_i}\left(\mathbf{D}_i^\top \widetilde{\mathbf{R}}_i\right), \tag{14}$$

where $\mathcal{H}_{s_i}$ retains the s_i largest-magnitude coefficients in each column. For fixed coefficients, let

$$\widetilde{\mathbf{R}}_i \mathbf{S}_i^\top = \mathbf{P}_i \mathbf{\Lambda}_i \mathbf{Q}_i^\top. \tag{15}$$

The dictionary update is the orthogonal Procrustes solution

$$\mathbf{D}_i \leftarrow \mathbf{P}_i \mathbf{Q}_i^\top. \tag{16}$$

After optimization, we return to the original weight space using

$$\mathbf{A}_i = \mathbf{L}_i^{-\top} \mathbf{D}_i, \qquad \widehat{\mathbf{W}}_i = \mathbf{B} + \mathbf{A}_i \mathbf{S}_i. \tag{17}$$

During BCD, the dictionary is warm-started across outer iterations, allowing the residual representation to adapt continuously to the changing base.

3.4 Model-Wide Budget Allocation

The dimensions (k_j, s_j) determine how much storage is assigned to each residual projection. We do not introduce a new allocator; instead, we adapt the one-shot spectral allocation of COMPOT (Makhov et al., 2026b) to MoE residuals and keep the resulting budgets fixed throughout BCD.

Let $j \in \mathcal{J}$ index all expert residual projections across all layers, experts, and projection types. After initializing the shared bases, we form the initial residuals in the original weight space,

$$\mathbf{R}_j^{(0)} = \mathbf{W}_j - \mathbf{B}_j^{(0)}, \qquad \overline{\mathbf{R}}_j^{(0)} = \frac{\mathbf{R}_j^{(0)}}{\|\mathbf{R}_j^{(0)}\|_F}. \tag{18}$$

For each normalized residual, we compute

$$\overline{\mathbf{R}}_j^{(0)} = \mathbf{U}_j \operatorname{diag}(\sigma_{j,1}, \ldots, \sigma_{j,q_j}) \mathbf{V}_j^\top. \tag{19}$$

The singular values from all $j \in \mathcal{J}$ are pooled into a single set,

$$\mathcal{S} = \bigcup_{j \in \mathcal{J}} \{\sigma_{j,\ell}\}_{\ell=1}^{q_j}, \tag{20}$$

and the smallest values are removed globally until the target residual budget is reached. The number removed from each residual induces its individual compression ratio c_j. Thus, more spectrally redundant residuals receive stronger compression, while residuals with heavier spectral tails retain more capacity.

For OSDL, the resulting compression ratio is converted to (k_j, s_j) at a fixed dictionary-to-sparsity ratio k_j/s_j. The storage of a residual is

$$S_j(k_j, s_j) = b\left(d_{\text{in},j} k_j + s_j d_{\text{out},j}\right) + \left\lceil \frac{k_j d_{\text{out},j}}{8} \right\rceil, \tag{21}$$

where b is the number of bytes per floating-point value and the final term accounts for the bit-packed sparsity mask. We choose (k_j, s_j) to match the allocated residual budget while preserving the selected k/s ratio. The full-model compression target is converted to a residual budget after accounting for the shared bases and all unchanged model parameters.

In the default *global* allocation, $\mathcal{S}$ contains residuals from all experts, layers, and projection types. A uniform allocation instead assigns the same compression ratio to every residual. Whether MoE-specific structure warrants restricting the spectral pool by layer, projection type, or expert identity is not obvious; we study these alternatives in Section 4.2.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Models and compression. We evaluate BOLD-MoE on Mixtral-8×7B (Jiang et al., 2024), Phi-3.5-MoE (Abdin et al., 2024), and DeepSeekMoE-16B-Base (Dai et al., 2024), covering different MoE architectures and expert granularities. We compress the gate, up, and down projections of the expert MLPs while leaving attention, embeddings, normalization layers, routers, and the output head unchanged. Compression ratio (CR) is measured with respect to the full-model storage, including the shared bases, OSDL dictionaries, sparse coefficients and masks, and all unchanged parameters. The corresponding residual budget is determined before model-wide allocation. We evaluate CRs of 20%, 40%, and 60%.

Baselines. Our primary baseline is D^2-MoE (Gu et al., 2025), the closest shared-base residual-compression method. We additionally compare with MoE-SVD (Li et al., 2025), NAEE (Lu et al., 2024), and MoE-I^2 (Yang et al., 2024b) where results are available. All D^2-MoE results in the main paper are obtained from our rerun of the released implementation under the same evaluation pipeline as BOLD-MoE. Appendix B documents the reproduction procedure, the preprocessing corrections, and the original paper-reported values.

Evaluation protocol. We use 256 sequences of length 2,048 from the WikiText-103 training split (Merity et al., 2017a) to collect the activation statistics required for compression. We report perplexity on WikiText-2 (Merity et al., 2017b), Penn Treebank (PTB) (Marcus et al., 1993), and C4 (Raffel et al., 2020). Zero-shot evaluation uses OpenBookQA (Mihaylov et al., 2018), ARC-Easy and ARC-Challenge (Clark et al., 2018), WinoGrande (Sakaguchi et al., 2020), HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), and MathQA (Amini et al., 2019); we report their mean accuracy as *Avg.* We additionally evaluate MMLU (Hendrycks et al., 2021) in Appendix A.4. All methods are evaluated without retraining. Unless stated otherwise, ablations use Mixtral-8×7B at 40% CR. Unless an ablation explicitly studies budget allocation, all OSDL ablations use uniform allocation; the final BOLD-MoE results use the global dynamic allocation of Section 3.4.

4.2 ABLATION STUDIES

Does alternating base–residual optimization help with SVD? We first isolate the effect of alternating optimization while retaining the SVD-based residual compression of D^2-MoE. Starting from the same one-shot decomposition, we repeatedly recompute the SVD-compressed residuals using equation 11 and update the shared base using equation 8. Figure 2 shows a steady reduction in both WikiText-2 and C4 perplexity relative to the one-shot initialization. Thus, alternating optimization improves the original SVD-based shared-base compression without changing the residual representation.

Does sparse dictionary learning improve residual compression? We next isolate the residual representation itself. After constructing the shared base, we keep it fixed and directly approximate the resulting expert residuals with SVD or OSDL, without any subsequent base update. For OSDL, we compare uniform allocation with the one-shot global dynamic allocation of Section 3.4, where residual projections from all experts, layers, and projection types compete under the same model-wide budget. Table 1 therefore separates the effect of residual geometry from base re-optimization. With $k/s = 2$, OSDL with uniform allocation reduces WikiText-2/C4 perplexity from 5.47/9.69 to 5.06/8.90 and increases Avg. accuracy from 56.3% to 58.3%. One-shot global allocation further improves these values to 5.01/8.84 and 58.6%. Thus, the union-of-subspaces representation improves residual compression even with the shared base held fixed.

How should compression budgets be grouped? The global allocator in Section 3.4 pools spectra from all residual projections. We next test whether restricting this competition to smaller groups is beneficial for MoE models. Table 2 compares global allocation with per-layer and per-projection-type grouping under the same parameter budget. Neither grouping improves consistently over global allocation, which achieves the best WikiText-2 and C4 perplexity. We additionally investigate MoE-specific groupings based on expert identity and its combinations with layer and projection structure

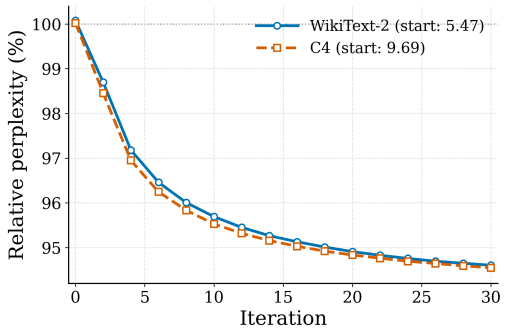


Figure 2: Alternating optimization with SVD residuals on Mixtral-8×7B at 40% CR. Perplexities are normalized to the one-shot D^2-MoE initialization at iteration 0; absolute starting values are shown in the legend. Repeated residual and shared-base updates steadily improve both WikiText-2 and C4 perplexity.

Table 1: Residual compression ablation on Mixtral-8×7B at 40% CR with a fixed shared base. OSDL uses 10 updates and $k/s = 2$.

Factorization	Allocation	WikiText-2↓	C4↓	Avg.↑
SVD	–	5.47	9.69	56.3
OSDL	Uniform	5.06	8.90	58.3
OSDL	Dynamic	**5.01**	**8.84**	**58.6**

Table 2: Grouping strategies for one-shot dynamic allocation on Mixtral-8×7B at 40% CR with $k/s = 3.4$ and 50 OSDL iterations.

Grouping	WikiText-2↓	PTB↓	C4↓	Avg.↑
Global	**4.93**	18.24	**8.65**	**59.0**
Per layer	4.95	18.61	8.66	58.9
Per projection type	4.98	21.02	8.70	58.9

in Appendix A.5, where we provide their precise definitions and complete results. These alternatives likewise provide no consistent improvement over global allocation. We therefore use global allocation in all remaining experiments.

How many OSDL updates are needed? We next examine how quickly OSDL converges when the shared base is fixed. Figure 4 shows the iteration sweep on Mixtral-8×7B at 40% CR with $k/s = 2$. Iteration 0 corresponds to the initial D^2-MoE decomposition before OSDL optimization. Most of the reduction in WikiText-2 and C4 perplexity is obtained within the first five updates, after which improvements become gradual. Thus, residual-only optimization quickly exhibits diminishing returns, motivating the later BOLD-MoE schedule that repeatedly revisits the shared base instead of spending many consecutive updates on a fixed residual target.

Dictionary size and coefficient sparsity. We next study the choice of dictionary size relative to coefficient sparsity. Starting from the initialized shared base, we keep the base fixed and compress the resulting expert residuals with OSDL for 50 iterations using uniform allocation, while varying k/s. The ratio k/s controls the trade-off between dictionary size and coefficient sparsity, with sparsity equal to $1 - s/k$. Figure 3 shows that quality improves rapidly from $k/s = 1$ and then remains nearly flat over the range $k/s = 2.6$–3.4. At $k/s = 3.4$, the coefficient matrices are 70.6% sparse while performance remains close to the best point in the sweep. We therefore use $k/s = 3.4$ in the main experiments, providing a highly sparse representation compatible with efficient sparse kernels such as cuSPARSE (NVIDIA Corporation, 2026). Appendix A.2 discusses the corresponding computational complexity and inference implications in more detail. The quality-optimal ratio is architecture dependent; additional results are reported in Appendix A.6.

How frequently should the shared base be updated? We finally study how the residual-optimization budget should be distributed within BCD. Each *outer* cycle consists of several OSDL *inner* iterations with the shared base fixed, followed by one update of the shared base. Each inner OSDL iteration alternates a sparse-code update with an orthogonal dictionary update. We fix the total number of OSDL inner iterations to 50 and vary how many are performed before each base update. Table 3 shows that frequent alternation is more effective than extensively optimizing the residuals against a fixed base. Moving from 50 OSDL iterations per outer cycle with a single base update to one OSDL iteration per cycle with 50 base updates improves WikiText-2/C4 perplexity from 4.94/8.64 to 4.74/8.34 and Avg. accuracy from 58.9% to 60.1%. We therefore use one OSDL iteration per outer cycle in the final method.

This ablation uses uniform allocation to isolate the optimization schedule; the final BOLD-MoE results in Table 4 use global dynamic allocation.

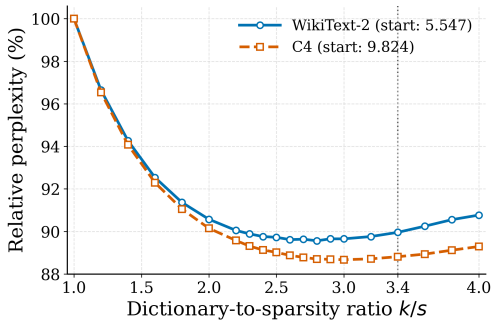


Figure 3: Sensitivity to the k/s ratio on Mixtral-8×7B at 40% CR.

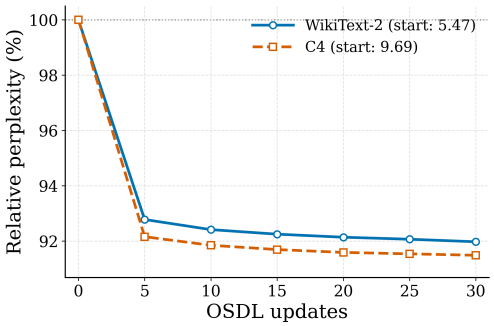


Figure 4: Fixed-base OSDL convergence on Mixtral-8×7B at 40% CR with $k/s = 2$. Perplexities are normalized to the D^2-MoE initialization at iteration 0.

Table 3: Frequency of shared-base updates on Mixtral-8×7B at 40% CR using uniform OSDL allocation. Each BCD cycle performs the indicated number of OSDL iterations with the shared base fixed, followed by one base update. The total number of OSDL iterations is fixed to 50 and $k/s = 3.4$.

OSDL steps/cycle	BCD cycles	WikiText-2↓	PTB↓	C4↓	Avg.↑	Time (h)
1	50	**4.74**	19.71	**8.34**	**60.1**	3.07
2	25	4.76	19.82	8.36	60.0	2.67
5	10	4.78	**19.27**	8.40	59.9	2.49
10	5	4.81	19.84	8.44	59.5	2.09
25	2	4.86	20.07	8.54	59.4	1.82
50	1	4.94	20.85	8.64	58.9	**1.79**

4.3 Main Results

Table 4 reports the final BOLD-MoE configuration across all three architectures and compression ratios; full task-wise zero-shot results are provided in Appendix A.3. BOLD-MoE consistently improves over D^2-MoE across all matched settings in WikiText-2, PTB, and C4 perplexity as well as average zero-shot accuracy. On Mixtral-8×7B at 40% CR, WikiText-2/PTB/C4 perplexity decreases from 5.47/22.69/9.69 to 4.69/17.55/8.27, while Avg. accuracy increases from 56.3% to 60.4%. The advantage becomes particularly pronounced under stronger compression: on Phi-3.5-MoE at 60% CR, BOLD-MoE reduces WikiText-2/PTB/C4 perplexity from 10.16/26.81/29.88 to 7.57/17.44/18.63 and improves Avg. accuracy from 43.3% to 49.6%. Improvements also persist on DeepSeekMoE-16B-Base despite its substantially different expert granularity; at 60% CR, WikiText-2/PTB/C4 perplexity decreases from 13.47/25.70/31.06 to 12.67/22.37/27.15. Thus, the improvement over the controlled D^2-MoE baseline persists across substantially different MoE architectures and compression levels. Relative to the broader baselines, BOLD-MoE also achieves the strongest perplexity in most evaluated settings, although individual exceptions remain; for example, MoE-SVD gives slightly lower WikiText-2 perplexity on Phi-3.5-MoE at 60% CR. The same trend extends to MMLU on Mixtral-8×7B: BOLD-MoE improves over D^2-MoE from 62.2/55.1/44.1 to 66.4/62.0/52.9 accuracy at 20/40/60% CR, respectively (Appendix A.4).

5 Conclusion and Limitations

We introduced **BOLD-MoE**, a post-training framework that treats the shared expert base and compressed expert-specific residuals as coupled variables rather than independent stages. Alternating base–residual optimization improves the original SVD-based decomposition, while orthogonal sparse dictionaries provide a more flexible residual representation. Across three MoE architectures and multiple compression ratios, BOLD-MoE consistently improves over the controlled D^2-MoE baseline in perplexity and zero-shot accuracy.

Table 4: Main compression results. Perplexity is reported on WikiText-2, PTB, and C4; Avg. is mean zero-shot accuracy over seven tasks. BOLD-MoE uses global dynamic OSDL allocation with $k/s = 3.4$ for all models.

CR	Method	Mixtral-8×7B				Phi-3.5-MoE				DeepSeekMoE-16B-Base			
		W2↓	PTB↓	C4↓	Avg.↑	W2↓	PTB↓	C4↓	Avg.↑	W2↓	PTB↓	C4↓	Avg.↑
0%	Original	3.98	12.99	6.78	63.0	3.48	8.43	8.22	62.0	6.38	9.47	9.82	56.0
20%	NAEE	4.77	16.09	8.89	58.0	–	–	–	–	9.44	15.02	15.34	53.0
	MoE-I^2	4.86	26.50	11.07	57.0	–	–	–	–	7.69	11.59	13.72	50.0
	MoE-SVD	4.86	19.42	8.98	58.0	4.58	10.13	9.93	**60.0**	6.92	10.48	11.99	**54.0**
	D^2-MoE	4.79	19.82	8.49	59.9	5.34	10.83	10.15	56.6	7.33	10.93	12.00	53.4
	BOLD-MoE	**4.17**	**15.12**	**7.53**	**62.1**	**4.46**	**9.73**	**9.05**	59.5	**6.92**	**10.36**	**10.83**	53.9
40%	NAEE	6.44	22.15	13.86	48.0	8.18	20.07	16.11	**57.0**	8.55	14.47	17.98	46.0
	MoE-I^2	6.74	60.45	22.44	49.0	7.46	20.95	20.95	45.0	9.73	15.75	19.75	45.0
	MoE-SVD	6.74	27.73	12.41	50.0	5.54	11.73	11.94	56.0	–	–	–	–
	D^2-MoE	5.47	22.69	9.69	56.3	6.56	13.39	13.45	53.3	8.67	13.46	15.79	49.4
	BOLD-MoE	**4.69**	**17.55**	**8.27**	**60.4**	**5.27**	**10.79**	**10.24**	56.9	**7.89**	**12.04**	**13.46**	**51.4**
60%	NAEE	11.43	47.28	31.16	36.0	–	–	–	–	23.20	49.89	48.63	38.0
	MoE-I^2	13.52	182.99	74.62	36.0	–	–	–	–	15.83	32.20	38.60	37.0
	MoE-SVD	13.52	130.26	39.54	37.0	**7.46**	20.95	21.85	49.0	–	–	–	–
	D^2-MoE	6.99	31.31	12.46	51.6	10.16	26.81	29.88	43.3	13.47	25.70	31.06	41.5
	BOLD-MoE	**5.82**	**23.60**	**10.23**	**55.2**	7.57	**17.44**	**18.63**	**49.6**	**12.67**	**22.37**	**27.15**	**42.8**

Several limitations remain. BOLD-MoE relies on routed calibration activations to estimate the activation-aware metric, so its quality may depend on how well the calibration data reflect deployment-time expert usage. The residual budgets (k_i, s_i) are allocated once before alternating optimization and remain fixed as the shared base changes. In addition, the preferred dictionary-to-sparsity ratio is architecture dependent, although we use a single $k/s = 3.4$ operating point in the main experiments. Finally, alternating optimization increases offline compression cost, and we do not evaluate end-to-end sparse MoE inference latency; realized runtime gains therefore depend on the available sparse backend and execution setting (Appendix A.2).

AI USE STATEMENT

We used generative AI tools only for language editing of the manuscript, including polishing the text, improving readability, and correcting grammar and style. All scientific content, methodology, experiments, results, analysis, and conclusions were produced and verified by the authors. We reviewed all AI-assisted edits and take full responsibility for the final content of the paper.

ETHICS STATEMENT

This work studies post-training compression of existing language models and does not involve human subjects, collection of personal or sensitive data, or the release of a new dataset. Our experiments use publicly available pretrained models and established language-modeling and downstream evaluation benchmarks. BOLD-MoE changes the parameterization and storage requirements of existing models rather than introducing new model capabilities; consequently, compressed models may retain the biases, limitations, and potential for misuse of their underlying pretrained models. At the same time, reducing model memory requirements can lower the hardware and energy barriers associated with deploying large Mixture-of-Experts models. We therefore recommend that compressed models be deployed subject to the same safety, licensing, and responsible-use considerations as their corresponding uncompressed models.

REPRODUCIBILITY STATEMENT

We provide the details required to reproduce BOLD-MoE throughout the paper and supplementary material. Section 3 specifies the optimization objective, alternating base–residual updates, and orthogonal sparse residual parameterization, while Section 4 describes the evaluated models, baselines, compression ratios, calibration procedure, and evaluation protocol. The appendix contains additional hyperparameter and k/s ablations, optimization-schedule experiments, and implementation details. We additionally document our reproduction of D^2-MoE in Appendix B, including the

original reported results, the discrepancies identified in the released implementation and our reproduction pipeline, the corresponding corrections, and the values used in the main paper. Source code, experiment launchers, configuration files, and evaluation scripts for the reported experiments are included in the supplementary material.

REFERENCES

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Weizhu Chen, Sam Ade Jacobs, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

Michal Aharon, Michael Elad, and Alfred M. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54 (11):4311–4322, 2006. doi: 10.1109/TSP.2006.881199.

Mengting Ai, Tianxin Wei, Yifan Chen, Zhichen Zeng, Ritchie Zhao, Girish Varatkar, Bita Darvish Rouhani, Xianfeng Tang, Hanghang Tong, and Jingrui He. ResMoE: Space-efficient compression of mixture of experts LLMs via residual restoration. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1–12. Association for Computing Machinery, 2025. doi: 10.1145/3690624.3709196.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2357–2367. Association for Computational Linguistics, 2019. doi: 10.18653/v1/N19-1245.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7432–7439, 2020. doi: 10.1609/aaai.v34i05.6239.

Patrick H. Chen, Hsiang-Fu Yu, Inderjit S. Dhillon, and Cho-Jui Hsieh. DRONE: Data-aware low-rank compression for large NLP models. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Xiaodong Chen, Mingming Ha, Zhenzhong Lan, Jing Zhang, and Jianguo Li. MoBE: Mixture-of-basis-experts for compressing MoE-based LLMs. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=8RV6H50OSf.

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

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. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1280–1297. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.70.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022. URL https://www.jmlr.org/papers/v23/21-0998.html.

Hao Gu, Wei Li, Lujun Li, Qiyuan Zhu, Mark G. Lee, Shengjie Sun, Wei Xue, and Yike Guo. Delta decompression for MoE-based LLMs compression. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 20497–20514. PMLR, 2025.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021.

Yen-Chang Hsu, Ting Hua, Sung-En Chang, Qian Lou, Yilin Shen, and Hongxia Jin. Language model compression with weighted low-rank factorization. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=uPv9Y3gmAI5.

Yongqi Huang, Peng Ye, Chenyu Huang, Jianjian Cao, Lin Zhang, Baopu Li, Gang Yu, and Tao Chen. DeRS: Towards extremely efficient upcycled mixture-of-experts models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10056–10066, 2025.

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. *arXiv preprint arXiv:2401.04088*, 2024.

Lujun Li, Qiyuan Zhu, Jiacheng Wang, Xiaoyu Qin, Wei Li, Hao Gu, Sirui Han, and Yike Guo. Submoe: Efficient mixture-of-expert llms compression via subspace expert merging. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 22994–23002, 2026. doi: 10.1609/aaai.v40i27.39464.

Wei Li, Lujun Li, Hao Gu, You-Liang Huang, Mark G. Lee, Shengjie Sun, Wei Xue, and Yike Guo. MoE-SVD: Structured mixture-of-experts LLMs compression via singular value decomposition. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 35209–35230. PMLR, 2025.

Xudong Lu, Qi Liu, Yuhui Xu, Aojun Zhou, Siyuan Huang, Bo Zhang, Junchi Yan, and Hongsheng Li. Not all experts are equal: Efficient expert pruning and skipping for mixture-of-experts large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6159–6172. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.334.

Vladimír Macko and Vladimír Boža. MACKO: Sparse matrix-vector multiplication for low sparsity. In *1st Workshop on Scaling Post-training for LLMs (SPOT), ICLR 2026*, 2026. URL https://openreview.net/forum?id=WnLnTrcMK9.

Denis Makhov, Dmitriy Shopkhoev, Magauiya Zhussip, Ammar Ali, and Stamatios Lefkimmiatis. CoSpaDi: Compressing LLMs via calibration-guided sparse dictionary learning. *arXiv preprint arXiv:2509.22075*, 2026a.

Denis Makhov, Dmitriy Shopkhoev, Magauiya Zhussip, Ammar Ali, Baher Mohammad, and Stamatios Lefkimmiatis. COMPOT: Calibration-optimized matrix procrustes orthogonalization for transformers compression. *arXiv preprint arXiv:2602.15200*, 2026b.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.

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

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

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2381–2391. Association for Computational Linguistics, 2018. doi: 10.18653/v1/D18-1260.

Stefan Nielsen, Rachel Teo, Laziz Abdullaev, and Tan Nguyen. Tight clusters make specialized experts. In *International Conference on Learning Representations*, 2025.

NVIDIA Corporation. cuSPARSE library. CUDA Toolkit Documentation, 2026. URL https://docs.nvidia.com/cuda/cusparse/.

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

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

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. SVD-LLM: Truncation-aware singular value decomposition for large language model compression. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=LNYIUouhdt.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024a.

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

Cheng Yang, Yang Sui, Jinqi Xiao, Lingyi Huang, Yu Gong, Yuanlin Duan, Wenqi Jia, Miao Yin, Yu Cheng, and Bo Yuan. MoE-i^2: Compressing mixture of experts models through inter-expert pruning and intra-expert low-rank decomposition. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 10456–10466. Association for Computational Linguistics, 2024b. doi: 10.18653/v1/2024.findings-emnlp.612.

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

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-1472.

A Additional Analysis and Results

A.1 Monotonicity of Alternating Base–Residual Optimization

We first describe the numerical stabilization used for the activation metric and then show that the alternating updates are non-increasing in the resulting reconstruction objective.

Gram-matrix stabilization. For each expert, we accumulate

$$\mathbf{G}_i^{\mathrm{raw}} = \mathbf{X}_i^\top \mathbf{X}_i. \tag{22}$$

Because routed activation statistics can be rank deficient, we symmetrize $\mathbf{G}_i^{\mathrm{raw}}$ and, when it is not positive definite, apply a small eigenvalue floor before Cholesky factorization. Let

$$\mathbf{G}_i^{\mathrm{raw}} = \mathbf{Q}_i \operatorname{diag}(\boldsymbol{\lambda}_i) \mathbf{Q}_i^\top. \tag{23}$$

We set

$$\widetilde{\boldsymbol{\lambda}}_i = \begin{cases} \max(\boldsymbol{\lambda}_i, \epsilon), & \min(\boldsymbol{\lambda}_i) \le 0, \\ \boldsymbol{\lambda}_i, & \text{otherwise}, \end{cases} \tag{24}$$

and reconstruct

$$\mathbf{G}_i = \mathbf{Q}_i \operatorname{diag}\left(\widetilde{\boldsymbol{\lambda}}_i\right) \mathbf{Q}_i^\top. \tag{25}$$

We use $\epsilon = 10^{-5}$ in all experiments and compute $\mathbf{G}_i = \mathbf{L}_i \mathbf{L}_i^\top$ by Cholesky factorization. The same stabilized matrix is used throughout residual compression and the shared-base update.

We now consider the objective

$$J\left(\mathbf{B}, \{\widehat{\mathbf{R}}_i\}_{i=1}^N\right) = \sum_{i=1}^N \left\| \mathbf{L}_i^\top \left(\mathbf{W}_i - \mathbf{B} - \widehat{\mathbf{R}}_i\right) \right\|_F^2. \tag{26}$$

Residual block. For a fixed base $\mathbf{B}^{(t)}$, define

$$\widetilde{\mathbf{R}}_i^{(t)} = \mathbf{L}_i^\top \left(\mathbf{W}_i - \mathbf{B}^{(t)}\right). \tag{27}$$

With the OSDL parameterization $\widehat{\mathbf{R}}_i = \mathbf{L}_i^{-\top} \mathbf{D}_i \mathbf{S}_i$, the corresponding residual objective is

$$\left\| \widetilde{\mathbf{R}}_i^{(t)} - \mathbf{D}_i \mathbf{S}_i \right\|_F^2. \tag{28}$$

For fixed orthogonal $\mathbf{D}_i$, the hard-thresholding update

$$\mathbf{S}_i \leftarrow H_{s_i}\left(\mathbf{D}_i^\top \widetilde{\mathbf{R}}_i^{(t)}\right) \tag{29}$$

is the exact minimizer under the column-wise sparsity constraint. For fixed $\mathbf{S}_i$, if

$$\widetilde{\mathbf{R}}_i^{(t)} \mathbf{S}_i^\top = \mathbf{P}_i \boldsymbol{\Lambda}_i \mathbf{Q}_i^\top, \tag{30}$$

then

$$\mathbf{D}_i \leftarrow \mathbf{P}_i \mathbf{Q}_i^\top \tag{31}$$

is the exact orthogonal Procrustes minimizer. Consequently, every sparse-code and dictionary update is non-increasing in the residual objective, and therefore

$$J\left(\mathbf{B}^{(t)}, \{\widehat{\mathbf{R}}_i^{(t+1)}\}\right) \le J\left(\mathbf{B}^{(t)}, \{\widehat{\mathbf{R}}_i^{(t)}\}\right). \tag{32}$$

Base block. For fixed compressed residuals $\{\widehat{\mathbf{R}}_i^{(t+1)}\}_{i=1}^N$, define

$$\mathbf{C}_i^{(t+1)} = \mathbf{W}_i - \widehat{\mathbf{R}}_i^{(t+1)}. \tag{33}$$

The base block is

$$\min_{\mathbf{B}} \sum_{i=1}^N \left\| \mathbf{L}_i^\top \left(\mathbf{C}_i^{(t+1)} - \mathbf{B}\right) \right\|_F^2. \tag{34}$$

Using $\mathbf{G}_i = \mathbf{L}_i \mathbf{L}_i^\top$, its gradient is

$$2 \sum_{i=1}^N \mathbf{G}_i \left(\mathbf{B} - \mathbf{C}_i^{(t+1)}\right). \tag{35}$$

Setting it to zero gives

$$\left(\sum_{i=1}^{N} \mathbf{G}_i\right) \mathbf{B}^{(t+1)} = \sum_{i=1}^{N} \mathbf{G}_i \mathbf{C}_i^{(t+1)}, \tag{36}$$

which is exactly equation 8. Thus,

$$J\left(\mathbf{B}^{(t+1)}, \{\widehat{\mathbf{R}}_i^{(t+1)}\}\right) \leq J\left(\mathbf{B}^{(t)}, \{\widehat{\mathbf{R}}_i^{(t+1)}\}\right). \tag{37}$$

Combining the residual and base blocks yields

$$J^{(t+1)} \leq J^{(t)}. \tag{38}$$

Since $J \geq 0$, the sequence of objective values converges. Because the OSDL constraints make the overall problem non-convex, this argument does not imply convergence to a globally optimal factorization.

A.2 Computational Complexity and Inference Implications

BOLD-MoE represents an expert projection as

$$\widehat{\mathbf{W}}_i = \mathbf{B} + \mathbf{A}_i \mathbf{S}_i, \tag{39}$$

where $\mathbf{A}_i \in \mathbb{R}^{d_{\text{in}} \times k_i}$ and every column of $\mathbf{S}_i \in \mathbb{R}^{k_i \times d_{\text{out}}}$ contains at most s_i nonzeros.

Residual-factor complexity. For one input vector, evaluating the OSDL residual $\mathbf{A}_i \mathbf{S}_i$ requires

$$C_{\text{OSDL},i} = d_{\text{in}} k_i + s_i d_{\text{out}} \tag{40}$$

multiplications when sparse coefficient execution is exploited. By comparison, a rank-r_i low-rank residual requires

$$C_{\text{SVD},i} = r_i \left(d_{\text{in}} + d_{\text{out}}\right). \tag{41}$$

This is the same dense-dictionary–sparse-coefficient operator studied by CoSpaDi (Makhov et al., 2026a). At matched storage, sparse dictionary and low-rank representations therefore have comparable arithmetic complexity, while their realized latency depends on sparsity, projection dimensions, indexing overhead, batching, and the available GPU kernels.

The main BOLD-MoE operating point uses $k/s = 3.4$, corresponding to

$$1 - \frac{s}{k} = 1 - \frac{1}{3.4} \approx 70.6\% \tag{42}$$

coefficient sparsity. This regime is compatible with sparse matrix multiplication backends such as cuSPARSE (NVIDIA Corporation, 2026). At lower sparsity, MACKO (Macko & Boža, 2026) provides a specialized matrix–vector implementation and reports substantial acceleration over dense execution at approximately 50% sparsity. CoSpaDi provides complementary evidence that the same sparse-dictionary operator can benefit from such hardware-specific sparse kernels (Makhov et al., 2026a).

Effect of the shared base. The complete BOLD-MoE projection additionally contains the dense shared base. For q routed experts receiving the same projection input, a backend that reuses the shared-base multiplication has theoretical cost

$$C_{\text{BOLD}} = d_{\text{in}} d_{\text{out}} + \sum_{i=1}^{q} \left(d_{\text{in}} k_i + s_i d_{\text{out}}\right), \tag{43}$$

instead of

$$C_{\text{dense}} = q\, d_{\text{in}} d_{\text{out}} \tag{44}$$

for q independent dense experts. Such reuse is natural for the gate and up projections, which receive the same token representation before expert-specific computation. For the down projection, expert inputs have already diverged, so the dense base multiplication cannot generally be shared in the same way.

We therefore view equation 40–equation 43 as evidence of the *potential* computational benefit rather than as a claim of backend-independent end-to-end acceleration. The present work evaluates compression quality and storage; a fully optimized MoE runtime that jointly exploits shared-base reuse and sparse residual kernels is left for future work.

Table 5: Detailed zero-shot results for the controlled shared-base comparison. All accuracy values are reported in percent. Avg. is the arithmetic mean over the seven tasks.

Model	Method	CR	OpenB.	ARC-e	ARC-c	WinoG.	HellaS.	PIQA	MathQA	Avg.↑
Mixtral-8×7B	Original	0%	36.0	84.0	57.0	76.0	65.0	82.0	43.0	63.0
	D^2-MoE	20%	34.0	80.6	51.7	75.1	60.4	80.2	37.7	59.9
	BOLD-MoE	20%	33.8	82.9	54.4	76.6	63.6	82.3	41.5	62.1
	D^2-MoE	40%	31.0	77.0	45.6	72.9	56.2	77.7	33.9	56.3
	BOLD-MoE	40%	33.8	80.9	51.4	75.7	61.1	80.9	39.0	60.4
	D^2-MoE	60%	27.8	69.7	38.1	70.9	50.2	73.9	30.4	51.6
	BOLD-MoE	60%	29.2	76.4	43.7	72.7	54.5	76.4	33.4	55.2
Phi-3.5-MoE	Original	0%	40.0	77.0	56.0	76.0	68.0	79.0	38.0	62.0
	D^2-MoE	20%	36.6	69.5	49.0	71.0	62.2	75.6	32.6	56.6
	BOLD-MoE	20%	38.6	73.3	52.5	74.2	66.0	77.3	35.0	59.5
	D^2-MoE	40%	35.2	65.5	44.9	69.9	54.5	73.4	29.6	53.3
	BOLD-MoE	40%	35.8	71.4	48.9	72.7	62.6	75.7	31.1	56.9
	D^2-MoE	60%	25.8	51.9	33.0	63.0	40.2	65.5	23.7	43.3
	BOLD-MoE	60%	29.2	63.2	41.2	68.8	50.2	68.9	25.6	49.6
DeepSeekMoE-16B-Base	Original	0%	32.0	76.0	44.0	70.0	58.0	79.0	31.0	56.0
	D^2-MoE	20%	30.2	74.6	41.8	69.8	52.4	75.2	30.1	53.4
	BOLD-MoE	20%	32.2	75.3	42.5	70.3	55.3	77.6	30.8	53.9
	D^2-MoE	40%	28.2	69.8	36.2	67.6	45.0	72.0	27.2	49.4
	BOLD-MoE	40%	30.0	72.4	39.4	69.0	49.7	74.4	28.4	51.4
	D^2-MoE	60%	20.8	57.4	28.1	61.0	34.7	64.1	24.1	41.5
	BOLD-MoE	60%	22.8	62.4	31.7	63.3	37.7	67.4	26.0	42.8

Table 6: MMLU accuracy on Mixtral-8×7B.

CR	Method	MMLU↑
0%	Original	67.9
20%	D^2-MoE	62.2
	BOLD-MoE	**66.4**
40%	D^2-MoE	55.1
	BOLD-MoE	**62.0**
60%	D^2-MoE	44.1
	BOLD-MoE	**52.9**

A.3 Full Task-Wise Main Results

The main paper reports average zero-shot accuracy over seven tasks. Table 5 expands the controlled comparison into the individual benchmarks: OpenBookQA, ARC-Easy, ARC-Challenge, WinoGrande, HellaSwag, PIQA, and MathQA. We report the original model together with D^2-MoE and BOLD-MoE, since these form the controlled shared-base comparison used throughout the paper.

A.4 Additional MMLU Evaluation

To complement the seven zero-shot benchmarks used in the main evaluation, we additionally evaluate MMLU (Hendrycks et al., 2021). Table 6 reports the controlled Mixtral-8×7B comparison. The advantage of BOLD-MoE persists at every compression ratio and becomes particularly large under stronger compression: at 60% CR, MMLU accuracy increases from 44.1% for D^2-MoE to 52.9% for BOLD-MoE.

A.5 MoE-Specific Dynamic-Allocation Grouping

The default allocator pools residual spectra globally across all layers, projection types, and experts. We additionally consider several restricted allocation pools:

Table 7: Complete dynamic-allocation grouping ablation on Mixtral-8×7B at 40% CR with $k/s = 3.4$. All strategies use the same residual-storage budget.

Grouping	WikiText-2↓	PTB↓	C4↓	Avg.↑
Global	**4.93**	18.24	**8.65**	59.0
Expert ID (global)	**4.93**	**17.63**	**8.65**	59.0
Per layer	4.95	18.61	8.66	58.9
Per layer–expert	4.94	18.24	8.66	**59.1**
Per layer–projection	4.99	21.06	8.72	59.0
Per projection–expert	4.98	20.47	8.71	58.9
Per projection type	4.98	21.02	8.70	58.9

- **Expert ID (global):** each expert index forms one group across all layers and projection types;
- **Per layer:** all experts and projection types within a layer share one budget;
- **Per layer–expert:** each $(\text{layer}, \text{expert})$ pair forms a group across projection types;
- **Per layer–projection:** each $(\text{layer}, \text{projection})$ pair forms a group across experts;
- **Per projection–expert:** each $(\text{projection}, \text{expert ID})$ pair forms a group across layers;
- **Per projection type:** each of gate, up, and down forms one model-wide group across layers and experts.

All variants use exactly the same global residual-storage budget; they differ only in which residuals are allowed to compete with one another. Table 7 shows that none of the MoE-specific groupings consistently improves over unrestricted global allocation. The best WikiText-2/C4 result is obtained by global allocation, while the largest Avg. difference among the alternatives is only 0.1 percentage points.

A.6 Architecture Dependence of the Dictionary-to-Sparsity Ratio

The ratio k/s controls the trade-off between dictionary capacity and coefficient sparsity. Larger values increase coefficient sparsity according to $1 - s/k$, but may eventually reduce approximation quality. The preferred value is not universal across architectures.

Mixtral. For Mixtral-8×7B at 40% CR, the fixed-base sweep is broad and flat: the best WikiText-2 perplexity is obtained around $k/s = 2.8$, the best C4 perplexity around $k/s = 3.0$, and the best average zero-shot accuracy around $k/s = 3.2$. The hardware-oriented $k/s = 3.4$ setting remains essentially on this plateau. For example, relative to the $k/s = 2.2$ point, moving to $k/s = 3.4$ changes WikiText-2 from 4.995 to 4.990, C4 from 8.800 to 8.725, and Avg. from 58.40% to 59.01%. Thus, increasing coefficient sparsity to 70.6% causes no quality loss in this Mixtral setting.

DeepSeekMoE. DeepSeekMoE-16B-Base is more sensitive. Table 8 compares the quality-oriented $k/s = 2.2$ configuration with the common $k/s = 3.4$ operating point used in the main experiments. We report the relative perplexity increase

$$\Delta_{\text{rel}} = 100 \left(\frac{\text{PPL}_{3.4}}{\text{PPL}_{2.2}} - 1 \right), \tag{45}$$

together with the absolute change in Avg. accuracy. The gap increases with compression strength: at 60% CR, moving from $k/s = 2.2$ to 3.4 raises WikiText-2/C4 perplexity by 10.8%/12.8% and reduces Avg. accuracy by 1.7 points.

These results explain why we use $k/s = 3.4$ as a single deployment-oriented operating point rather than claiming it is universally quality-optimal. On Mixtral, the quality curve is sufficiently flat that the additional sparsity is effectively free, whereas DeepSeekMoE exhibits a clearer quality–sparsity trade-off, particularly under severe compression. A matched k/s sweep for Phi-3.5-MoE would be required to characterize the same trade-off for that architecture.

Table 8: Sensitivity to the dictionary-to-sparsity ratio. Relative perplexity change compares $k/s = 3.4$ against $k/s = 2.2$; positive values denote degradation. Mixtral uses the fixed-base 50-update sweep, whereas DeepSeekMoE uses the corresponding BOLD-MoE runs, so the two model blocks should be interpreted as within-model sensitivity rather than a strictly matched cross-model benchmark.

Model	CR	WikiText-2			C4			Avg.		
		2.2	3.4	Δ_{rel}	2.2	3.4	Δ_{rel}	2.2	3.4	Δ (pp)
Mixtral-8×7B	40%	4.995	4.990	-0.1%	8.800	8.725	-0.9%	58.40	59.01	$+0.61$
DeepSeekMoE-16B-Base	20%	6.84	6.92	$+1.2\%$	10.65	10.83	$+1.7\%$	54.8	53.9	-0.9
	40%	7.66	7.89	$+3.0\%$	12.86	13.46	$+4.7\%$	51.9	51.4	-0.5
	60%	11.44	12.67	$+10.8\%$	24.06	27.15	$+12.8\%$	44.5	42.8	-1.7

Table 9: Published and reproduced D^2-MoE results. Avg. denotes mean zero-shot accuracy in percent. Only settings available in both evaluations are shown.

Model	CR	Source	WikiText-2↓	PTB↓	C4↓	Avg.↑
Mixtral-8×7B	40%	Published	5.28	20.54	10.10	57.0
		Ours	5.47	22.69	9.69	56.3
	60%	Published	6.46	23.63	12.76	52.0
		Ours	6.99	31.31	12.46	51.6
Phi-3.5-MoE	40%	Published	6.07	13.79	14.01	60.0
		Ours	6.56	13.39	13.45	53.3
DeepSeekMoE-16B-Base	20%	Published	6.84	11.10	11.88	54.0
		Ours	7.33	10.93	12.00	53.4
	40%	Published	7.93	14.07	15.18	49.0
		Ours	8.67	13.46	15.79	49.4
	60%	Published	11.67	27.73	27.63	41.0
		Ours	13.47	25.70	31.06	41.5

B Reproduction of D^2-MoE

D^2-MoE (Gu et al., 2025) is our primary shared-base baseline. Because our reproduced results differ from some values reported in the original paper, we describe the reproduction procedure here for transparency. We use the authors' released implementation and preserve its overall pipeline: expert routing statistics and Fisher information are used to construct the shared base, followed by SVD compression of the expert-specific residuals. For the controlled comparisons in the main paper, we do not prune the shared base to isolate the effect of residual compression and joint base-residual optimization, and choose the residual budget to match the target full-model compression ratio.

During reproduction, we identified several preprocessing and accounting details that materially affect the result. In particular, we corrected the causal-language-model preprocessing used for Fisher estimation, excluded padded tokens when computing expert routing frequencies, and standardized the residual-retention convention while explicitly verifying the resulting model storage. We otherwise preserve the released shared-base construction, including its combination of Fisher statistics and routing frequency. The same corrected statistics, initialization, compression accounting, and evaluation pipeline are used for both D^2-MoE and BOLD-MoE.

Table 9 compares the published D^2-MoE results with our reruns for settings available in both evaluations. Some differences remain and are not uniform across metrics. We do not attribute all of them to the preprocessing corrections above, since exact reproduction can additionally depend on software versions, checkpoints, data preprocessing, and evaluation details. We therefore use our rerun values in the main paper to provide a controlled comparison with BOLD-MoE, while reporting the original values here for reference.

To facilitate reproducibility, we accompany the submission with the source code, experiment configurations, and evaluation scripts used for the reported results. The code will be released publicly upon acceptance.