GS-MoE: Training-Free MoE Compression via Groupwise Subspace Sharing and Pruning

Anonymous authors
Paper under double-blind review

Abstract

Mixture-of-Experts (MoE) models reduce computation through sparse activation, but storing their expert parameters remains a major memory bottleneck. We introduce GS-MoE, a training-free compression framework that addresses redundancy across and within experts through two compression axes: shared-subspace rank and expert intermediate width. Groupwise low-rank representations capture shared structure across experts while retaining expert-specific projections, and Nyström-based neuron compression reduces intermediate width within each expert. For fixed shared subspaces, we derive per-expert reconstruction-error reductions from the reconstruction objective and use them to guide expert grouping. Budgets are allocated hierarchically using a gradient-free layer-importance score to guide the allocation of rank and width budgets across layers, which eigenvalues and leverage then distribute as non-uniform ranks across groups and intermediate widths across experts. Experiments demonstrate the complementary benefits of compression along both axes, particularly at high compression rates on fine-grained MoE models. On DeepSeek-MoE-16B-Base, at a 60% reduction in total model parameters, GS-MoE retains approximately 90% of the uncompressed model's average reasoning performance, compared with 73% for D^2-MoE, while achieving 20% lower out-of-domain perplexity.

1 Introduction

Mixture-of-Experts (MoE) models increase parameter capacity while limiting computation by activating only a subset of experts for each token (Rajbhandari et al., 2022; DeepSeek-AI et al., 2025). Sparse activation, however, does not eliminate the storage cost of inactive experts. The resulting expert parameter footprint remains a major obstacle to deployment, motivating compression methods that reduce memory requirements while preserving model quality (Li et al., 2025b; Gu et al., 2025).

Expert compression must account for both structure shared across experts and redundancy within each expert. Shared-factor methods such as MoE-SVD address the former with a common low-rank factor across experts where D^2-MoE stores a shared base with compressed expert-specific residuals (Li et al., 2025b; Gu et al., 2025). These parameterizations impose different constraints on how shared and expert-specific information is represented. In particular, MoE-SVD requires the corresponding expert factors to use one common subspace; directions outside that subspace cannot be recovered by changing the expert-specific projections alone. Intermediate-neuron pruning addresses a different limitation by reducing expert width (He et al., 2026b;a), but pruning alone still stores separate weight representations for the retained neurons of different experts. Thus, a fixed shared subspace restricts where expert-specific variation can be represented, whereas width reduction alone does not encode common structure through shared parameters. This motivates a representation that allows subspaces to vary across expert groups and intermediate widths to vary across experts, with both capacities controlled under a common parameter budget.

We introduce **GS-MoE**, a training-free framework that compresses MoE parameters along shared-subspace rank and expert intermediate width. Group-shared projections represent common structure across experts, while expert-specific projections express individual transformations within each group's subspaces. This makes the scope of parameter sharing explicit rather than fixing a single basis for the entire layer. Ridge-leverage neuron selection reduces intermediate width before the up projections are factorized. We estimate the shared up projections using a group-level surrogate and

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.

fit the expert-specific projections using each expert's input statistics. The down reconstruction objective admits a closed-form solution before width compression; Nyström reconstruction (Williams & Seeger, 2000) then maps the retained hidden activations to the down projection's shared coordinates. Group rank and expert width therefore control two dimensions of the same compressed representation: the capacity of shared subspaces and the size of expert-specific intermediate activations.

GS-MoE determines which experts share a subspace and how capacity is distributed under a parameter budget using calibration statistics. For fixed neuron selections and candidate shared projections, fitting the expert-specific projections expresses the reconstruction objective as a constant minus per-expert error reductions, which guide expert grouping. A gradient-free layer-importance score based on normalized, gate-weighted expert output magnitudes (Lasby et al., 2026) guides rank and width budgets across layers. Within each layer, neuron leverage scores determine expert widths, and eigenvalue-based scores and parameter costs guide group ranks. The rank scores reflect the up surrogate and the down reconstruction objective before width compression. This allocation adapts the compressed representation at the layer, group, and expert levels without post-compression training.

Experiments demonstrate the complementary benefits of compression along both axes, particularly at high compression rates on fine-grained MoE models. On DeepSeek-MoE-16B-Base, at 60% compression, GS-MoE retains 90% of the uncompressed model's average reasoning performance, compared with 73% for D^2-MoE, while achieving 20% lower out-of-domain perplexity. Our contributions are:

- We introduce GS-MoE, a training-free framework that compresses expert parameters along shared-subspace rank and intermediate width, addressing shared structure across experts and redundancy within experts.
- We derive per-expert reconstruction-error reductions for fixed neuron selections and shared projections, providing an error-guided criterion for expert grouping.
- We allocate capacity across layers, groups, and experts using a normalized expert-output score, eigenvalue-based rank scores, and neuron leverage scores under a parameter budget.
- We demonstrate the benefit of compression along both axes and improved reasoning retention and out-of-domain perplexity over D^2-MoE at 60% compression on DeepSeek-MoE-16B-Base.

2 Related Work

2.1 Pruning and Merging of Mixture-of-Experts

Through sparse activation the MoE architecture effectively limits computational demand but introduces significant parameter overhead and redundancy making MoE compression techniques vital. These techniques generally operate through one of expert merging, expert pruning, neuron pruning, or decomposition. Expert merging methods reduce parameters by consolidating similar experts: HC-SMoE (Chen et al., 2025) uses hierarchical clustering to merge similar experts without retraining while DERN (Zhou et al., 2025) redistributes neuron-level segments from pruned experts to compatible retained experts before merging them. Not All Experts are Equal (Lu et al., 2024) introduces expert pruning which reduces parameter overhead through whole expert removal; REAP (Lasby et al., 2026) builds on this by applying router-weighted output norms to score expert prunability. Further, REAP (Lasby et al., 2026) identifies that merging-based methods incur significant error that pruning-based methods do not. Rather than discarding entire modules, finer-grained frameworks such as TENP (He et al., 2026b) and Less-Is-MoE (He et al., 2026a) compress intermediate expert widths, out-performing whole expert removal. However, pruning techniques rely solely on parameter deletion, leaving shared inter-expert structures unexploited. In contrast, our method avoids the distortion of expert merging and the finality of pure pruning by coupling group-shared subspaces that capture inter-expert redundancy with intra-expert width compression.

2.2 Low-Rank Decomposition and Parameter Sharing

Decomposition-based frameworks replace expert weight matrices with compact low-rank factors. Early decomposition-based approaches apply low-rank approximation in conjunction with expert

pruning (Yang et al., 2024). Alternative methods employ parameter sharing to exploit cross-expert redundancy. MoE-SVD (Li et al., 2025b) shares singular vectors from the most frequently activated experts and D^2-MoE (Gu et al., 2025) represents each expert as a shared base weight plus a low-rank expert-specific residual. TD-MoE stacks expert weights into a tensor for joint Tucker decomposition with adaptive three-dimensional rank allocation and multi-linear whitening (Xu et al., 2026). In each of these three methods, the shared component, whether a common factor, an additive base, or a set of mode factors, spans every expert in the layer. An expert whose weights diverge must absorb the mismatch in its expert-specific component; under MoE-SVD, directions outside the shared subspace cannot be recovered by any choice of expert-specific factors. GS-MoE shares subspaces within groups of experts, and it forms those groups to maximize the reconstruction-error reduction each expert obtains, so experts with dissimilar input or output subspaces are not forced onto a common basis. Additionally, GS-MoE adapts MoDeGPT (Lin et al., 2025), an intermediate-width compression method for dense LLMs, to its MoE compression framework. GS-MoE minimizes reconstruction error over width-reduced up projections, effectively folding an additional compression axis directly into the subspace.

Other methods relax the single-shared subspace constraint. LorExperts (Chakrabarti et al., 2026) keeps one full-precision dominant expert per cluster and represents the other members as low-rank corrections to it. MoBE (Chen et al., 2026) builds each expert's up and gate matrices from its own combination of basis matrices shared across the layer, fitted by gradient-based weight reconstruction, leaving the down projections uncompressed. Sub-MoE (Li et al., 2025a) utilizes k-means clustering to group experts clusters by cosine similarity, align clusters to a shared basis by SVD, then merge the clusters into a single expert. Our method uses balanced medoid clustering with k-means++ initialization to group experts using reconstruction-error reductions. MoLAE (Liu et al., 2025) is closest to GS-MoE's reparameterization through its group-shared latent projections with expert-specific transformations via stacked SVD. However, MoLAE forms arbitrary groups, and is forced to preserve the up projections entirely and restrict intermediate width to the latent rank forming compression factors of $\mathbb{R}^{m\times m}$, bounding its compression by the ratio of the two dimensions. GS-MoE retains and compresses every expert, including its down projections, and treats group rank and expert width as separate budgeted quantities allocated across layers, groups, and experts from calibration statistics, so its compression is set by the parameter budget rather than by a fixed dimension ratio or an uncompressed component.

3 METHOD

3.1 NOTATION

We consider one layer with E experts, model dimension d, and intermediate width d_{ff}; the layer index l is included only for layer-wise capacity allocation. In group-wise sharing, $e \in \mathcal{E}_g$ denotes that expert e belongs to group g, with G groups in total. We denote the original expert weights by $W_e^\star$, group-shared projection matrices by $P_g^\star$, and expert-specific projection matrices by $A_e^\star$, where $\star \in \{\uparrow, \downarrow\}$ indicates the up or down projections. For simplicity, we omit gated projections.

3.2 COMPRESSION ALONG RANK AND WIDTH

An FFN expert computes $f_e(x) = W_e^\downarrow \phi(W_e^\uparrow x)$, where $\phi(\cdot)$ is the nonlinear operation, $W_e^\uparrow \in \mathbb{R}^{d_{\text{ff}}\times d}$ and $W_e^\downarrow \in \mathbb{R}^{d\times d_{\text{ff}}}$. After compression with GS-MoE, we have the following computation for $e \in \mathcal{E}_g$:

$$\hat{f}_e(x) = P_g^\downarrow A_e^\downarrow \phi\big(A_e^\uparrow (P_g^\uparrow)^\top x\big)\,. \tag{1}$$

Here $P_g^\uparrow, P_g^\downarrow \in \mathbb{R}^{d\times r_g}$ are group-shared projections, where r_g is the latent dimension of group g. The expert-specific projections $A_e^\uparrow \in \mathbb{R}^{k_e\times r_g}$ and $A_e^\downarrow \in \mathbb{R}^{r_g\times k_e}$ retain k_e intermediate neurons for expert e. The input is first projected into the r_g-dimensional shared space by $(P_g^\uparrow)^\top$, then mapped to k_e intermediate activations by $A_e^\uparrow$. After the nonlinearity, $A_e^\downarrow$ maps these activations back to r_g dimensions, and $P_g^\downarrow$ restores the model dimension d. For gated experts, the gate and up projections use separate shared and expert-specific factors with the same r_g and k_e. The gate projection follows the same derivation as the up projection.

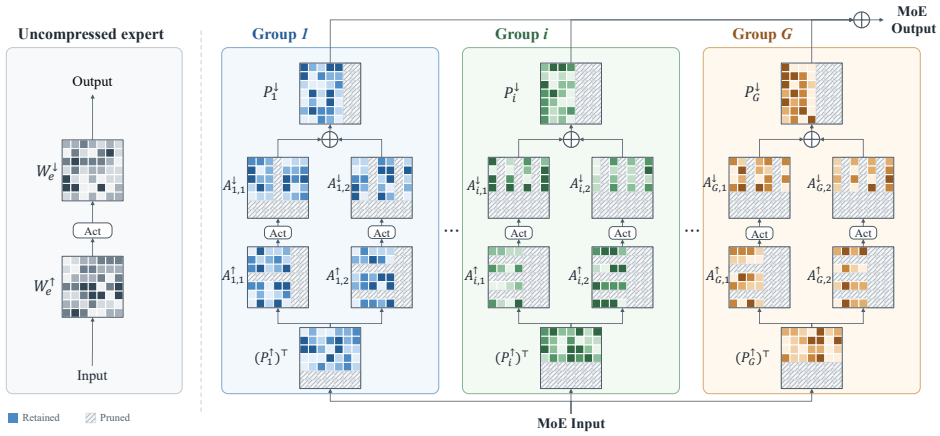


Figure 1: Overview of GS-MoE. Compared with an uncompressed expert (left), GS-MoE partitions experts into equally sized groups with shared input and output projections $P_g^{\uparrow}$ and $P_g^{\downarrow}$, while retaining expert-specific projections $A_{g,j}^{\uparrow}$ and $A_{g,j}^{\downarrow}$, where j indexes experts within group g, so $A_{g,j}^{\star}$ corresponds to $A_e^{\star}$ for the associated expert $e \in \mathcal{E}_g$. Group ranks and expert intermediate widths are allocated non-uniformly under a fixed parameter budget. Colored cells indicate retained parameters, and hatched regions indicate removed dimensions. Two experts per group are shown for illustration; routing weights and gated projections are omitted.

3.3 Groupwise Subspace Sharing and Pruning

We estimate shared projections and expert-specific projections separately, with neuron selection applied before up factorization and reconstruction applied after down factorization. The up projection requires a group-level surrogate, whereas the down projection admits an exact solution for its corresponding reconstruction objective before width compression.

Calibration and neuron selection. For each expert e, let h_e be its original intermediate hidden activation and $y_e = W_e^{\downarrow} h_e$ its output. We collect the second moments $C_e^x = \mathbb{E}_e[xx^\top]$, $C_e^h = \mathbb{E}_e[h_e h_e^\top]$, and $C_e^y = \mathbb{E}_e[y_e y_e^\top]$, where $\mathbb{E}_e$ averages over calibration inputs routed to expert e. Given total width $K = \sum_e k_e$, we select neurons by their ridge-leverage scores $q_{e,i} = [C_e^h (C_e^h + \eta I)^{-1}]_{ii}$, where $\eta > 0$. The top-K scores across experts determine k_e and the selection matrix $S_e \in \mathbb{R}^{d_{\text{ff}} \times k_e}$, whose columns are the corresponding columns of the identity matrix. The same S_e selects the up and gate neurons.

Neuron selection enforces a minimum retained width per expert.

Up projection: For fixed neuron selections, the up projection minimizes the reconstruction error of the selected intermediate pre-activations:

$$\min_{P_g^{\uparrow}, A_e^{\uparrow}} \sum_{e \in \mathcal{E}_g} \mathbb{E}_e \left\| S_e^\top W_e^{\uparrow} x - A_e^{\uparrow} (P_g^{\uparrow})^\top x \right\|_2^2, \quad (P_g^{\uparrow})^\top P_g^{\uparrow} = I_{r_g}. \tag{2}$$

The joint up problem does not, in general, admit a closed-form solution when expert input statistics differ. Although the expert-specific projections have closed-form solutions for a fixed shared projection, eliminating them leaves expert-dependent inverse terms that still depend on the shared projection. We therefore estimate the shared projection using a group-level surrogate and then fit $A_e^{\uparrow}$ using each expert's own input statistics. Let $C_g = \sum_{e \in \mathcal{E}_g} \tau_e C_e^x$, where $\tau_e = n_e / \sum_{j \in \mathcal{E}_g} n_j$ and n_e counts routed calibration assignments. Using the surrogate matrix $H_g^{\uparrow}$ defined in Section 3.5, the resulting shared and expert-specific projections are

$$\begin{aligned} P_g^{\uparrow} &= \operatorname{orth}\left(C_g^{-1/2} \operatorname{eig}_{r_g}(H_g^{\uparrow}) \right), \\ A_e^{\uparrow} &= S_e^\top W_e^{\uparrow} C_e^x P_g^{\uparrow} \left((P_g^{\uparrow})^\top C_e^x P_g^{\uparrow} \right)^{-1}. \end{aligned} \tag{3}$$

Here $\operatorname{eig}_r(\cdot)$ returns the leading r orthonormal eigenvectors, and $\operatorname{orth}(\cdot)$ orthonormalizes the columns. The expert-specific projections minimize the error for the chosen shared projection; the

shared projection is an approximation to the original objective. The surrogate construction and expert-specific projection solution are derived in Appendix A.1. The gate projection follows the same construction.

Down projection: The down projection minimizes the reconstruction error of the original expert outputs:

$$\min_{P_g^{\downarrow}, B_e} \sum_{e \in \mathcal{E}_g} \mathbb{E}_e \left\| y_e - P_g^{\downarrow} B_e h_e \right\|_2^2, \quad (P_g^{\downarrow})^\top P_g^{\downarrow} = I_{r_g}. \tag{4}$$

Here $B_e \in \mathbb{R}^{r_g \times d_{\text{ff}}}$ denotes the temporary projections before intermediate-width compression. The joint down problem admits a closed-form solution. Eliminating B_e reduces Obj. 4 to finding the leading eigenvectors of the summed output second moments. The optimal shared and temporary expert-specific projections are

$$P_g^{\downarrow} = \text{eig}_{r_g} \left(\sum_{e \in \mathcal{E}_g} C_e^y \right), \qquad B_e = (P_g^{\downarrow})^\top W_e^{\downarrow}. \tag{5}$$

This solution minimizes the objective before width compression. Appendix A.2 derives this solution.

Intermediate-width compression introduces a further approximation: the full hidden activation is reconstructed from the selected coordinates using Nyström approximation (Williams & Seeger, 2000), giving the final expert-specific projections:

$$A_e^{\downarrow} = B_e C_e^h S_e (S_e^\top C_e^h S_e)^\dagger \in \mathbb{R}^{r_g \times k_e}, \tag{6}$$

where † denotes the Moore–Penrose pseudoinverse; the reconstruction is derived in Appendix A.3. Thus, down-width reconstruction follows fitting the shared output projection, while up/gate selection precedes factorization. All statistics come from the original model.

3.4 Error-Guided Expert Grouping

We assign experts to the G groups by minimizing reconstruction error. For fixed neuron selections and candidate shared projections, $A_e^{\uparrow}$ and B_e take their optimal values for each candidate group. Writing $g = g(e)$ within each summand, the assignment satisfies

$$\begin{aligned} &\arg\min_{g(\cdot)} \sum_e \mathbb{E}_e \left[\| S_e^\top W_e^{\uparrow} x - A_e^{\uparrow} (P_g^{\uparrow})^\top x \|_2^2 + \| y_e - P_g^{\downarrow} B_e h_e \|_2^2 \right] \\ = &\arg\max_{g(\cdot)} \sum_e \underbrace{\left\{ \begin{array}{l} \text{tr}\Big[\left((P_g^{\uparrow})^\top C_e^x P_g^{\uparrow} \right)^{-1} (P_g^{\uparrow})^\top C_e^x (W_e^{\uparrow})^\top S_e S_e^\top W_e^{\uparrow} C_e^x P_g^{\uparrow} \Big] \\ + \text{tr}\big[(P_g^{\downarrow})^\top C_e^y P_g^{\downarrow} \big] \end{array} \right\}}_{J_e^g}. \end{aligned} \tag{7}$$

Here J_e^g is the reconstruction-error reduction after fitting $A_e^{\uparrow}$ and B_e, and the assignment satisfies the group-size constraints. The gate projection contributes an analogous term. Appendix A.4 provides the derivation. We use this error-reduction criterion to construct the pairwise assignment scores below, then fit shared projections to the resulting groups using Section 3.3.

Up/down balance. Neuron selection reduces the up term of Obj. 7. To balance the grouping scores, we multiply their down term by the retained pre-activation second-moment fraction of the up projections:

$$\beta_{\text{scale}} = \frac{\sum_{e=1}^{E} \text{tr}\big(S_e^\top W_e^{\uparrow} C_e^x (W_e^{\uparrow})^\top S_e \big)}{\sum_{e=1}^{E} \text{tr}\Big(W_e^{\uparrow} C_e^x (W_e^{\uparrow})^\top \Big)}. \tag{8}$$

For gated experts, we add the corresponding gate term to both the numerator and denominator, using the same neuron selections. This gives the error-reduction score for a reconstruction objective whose down error is weighted by β_{scale}. The scaling is used only for grouping; projection fitting and rank allocation retain the original C_e^y.

Group assignment. For experts $i, j \in \{1, \ldots, E\}$, let J_i^j be the score from Obj. 7, with the up/down balancing above, for expert i evaluated on the rank-r subspace fitted to expert j. We define the

pairwise dissimilarity matrix $D \in \mathbb{R}^{E\times E}$ by $D_{i,j} = 1 - J_i^j / J_i^i$, where J_i^i is the self-reconstruction score. Because the up subspace is fitted approximately, this normalization does not guarantee that $D_{i,j}$ lies in $[0, 1]$. Groups are then obtained by balanced medoid clustering on D with cluster sizes fixed to E/G and assignment solved exactly by the Jonker-Volgenant (Crouse, 2016) algorithm over medoids replicated into E/G slots. We initialize medoids using k-means++ seeding and alternate balanced assignment with medoid updates. The pairwise scores require E single-expert subspace fits and an $E \times E$ gain sweep.

3.5 Capacity Allocation

Within a layer. Assume each expert has n weight matrices, each containing $d \cdot d_{\text{ff}}$ parameters, with dimensions $d_{\text{ff}} \times d$ or $d \times d_{\text{ff}}$. Each matrix has its own shared and expert-specific projections, while all matrices use the same group rank r_g and retained width k_e. The compressed weights therefore contain $n \sum_g r_g c_g$ parameters, where $c_g = d + \sum_{e\in\mathcal{E}_g} k_e$.

For fixed groups and neuron selections, rank allocation uses the matrices underlying the projection solutions:

$$\begin{aligned} H_g^{\uparrow} &= C_g^{1/2} \left(\sum_{e\in\mathcal{E}_g} \tau_e (W_e^{\uparrow})^{\top} S_e S_e^{\top} W_e^{\uparrow} \right) C_g^{1/2}, \\ H_g^{\downarrow} &= \sum_{e\in\mathcal{E}_g} C_e^{y}. \end{aligned} \tag{9}$$

Here C_g and τ_e are defined in Section 3.3. The leading eigenvalues of $H_g^{\uparrow}$ measure the reduction in the up surrogate, whereas those of $H_g^{\downarrow}$ measure the reduction in Obj. 4 before width compression. For matrix $j \in \{1, \ldots, n\}$, let $H_g^{(j)}$ denote the corresponding construction using that matrix's weights and calibration statistics. We define $V_g(t) = \sum_{j=1}^{n} \sum_{i=1}^{t} \lambda_i(H_g^{(j)})$, with eigenvalues in descending order. This is a rank-allocation score, not the exact error reduction $\sum_e J_e^g$ in Section 3.4, since its up terms use the surrogate. Under the uniform-rank reference budget, we solve

$$\begin{aligned} \max_{\{r_g\}} \quad & \sum_g V_g(r_g) \\ \text{s.t.} \quad & \sum_g r_g c_g \le r \sum_g c_g, \qquad r_{\min} \le r_g \le r_{\max}, \quad r_g \in \mathbb{Z}. \end{aligned} \tag{10}$$

The common factor n cancels from the budget constraint. Widths are selected first using the neuron scores under the total budget K, fixing both c_g and the selections used in $H_g^{(j)}$ during rank allocation. We approximately solve Obj. 10 by bisecting a Lagrange multiplier for the parameter budget, following the multiplier-search approach to resource allocation (Patriksson, 2008), then greedily allocate the remaining budget. Appendix A.5 describes the discrete allocation steps.

Across layers. We now introduce the layer index ℓ. Using the gate-weighted expert output criterion of REAP (Lasby et al., 2026), normalized by input magnitude, we define

$$I_\ell = \sum_{e=1}^{E_\ell} \mathbb{E}_{\ell,e} \left[\frac{\pi_{\ell,e}(x) \| f_{\ell,e}(x) \|_2}{\|x\|_2} \right], \tag{11}$$

where $\pi_{\ell,e}(x)$ is the routing weight and $\mathbb{E}_{\ell,e}$ averages over inputs routed to that expert. Given a target parameter budget, we first choose the overall width retention and determine the corresponding rank budget, accounting for the shared projections. We then allocate rank and width across layers using I_ℓ, assigning less compression to more important layers. Because parameter count depends jointly on rank and width, we keep the width allocation fixed and rescale the layer ranks to match the total budget in the continuous relaxation. The resulting r_ℓ and K_ℓ guide the within-layer allocation of group ranks and expert widths. Integer rounding and unused allocation slack can change the achieved compression. Appendix A.6 gives the allocation and budget-correction details.

Table 1: Comparison of GS-MoE to other methods. Best results are in **bold**.

Ratio	Method	Wiki.↓	PTB↓	OpenB.	ARC-e	WinoG.	HellaS.	ARC-c	PIQA	MathQA	Avg.↑
DeepSeekMoE-16B-Base (64 experts, top-6 activated)											
0%	Original	6.38	9.47	0.32	0.75	0.70	0.58	0.44	0.79	0.31	0.56
20%	NAEE (2024)	9.44	15.02	0.32	0.71	0.66	0.55	0.40	0.77	0.29	0.53
	MoE-I^2 (2024)	7.69	11.59	0.26	0.71	0.68	0.49	0.38	0.73	0.29	0.51
	D^2-MoE (2025)	**6.84**	11.10	0.30	0.74	0.69	0.55	0.41	0.76	**0.31**	0.54
	GS-MoE (Ours)	7.35	**10.64**	**0.33**	**0.75**	**0.70**	**0.57**	**0.43**	**0.79**	0.30	**0.55**
40%	NAEE (2024)	8.55	14.47	0.23	0.67	0.67	0.41	0.32	0.69	0.26	0.46
	MoE-I^2 (2024)	9.73	15.75	0.23	0.64	0.66	0.41	0.31	0.68	0.26	0.46
	D^2-MoE (2025)	**7.93**	14.07	0.26	0.69	0.65	0.45	0.36	0.72	0.28	0.49
	GS-MoE (Ours)	10.10	**13.51**	**0.31**	**0.75**	**0.70**	**0.53**	**0.43**	**0.78**	**0.29**	**0.54**
60%	NAEE (2024)	23.20	49.89	0.17	0.49	0.58	0.33	0.24	0.61	0.23	0.38
	MoE-I^2 (2024)	15.83	32.20	0.17	0.48	0.58	0.32	0.23	0.61	0.22	0.37
	D^2-MoE (2025)	**11.67**	27.73	0.21	0.54	0.61	0.35	0.29	0.63	0.24	0.41
	GS-MoE (Ours)	17.55	**22.21**	**0.28**	**0.72**	**0.66**	**0.45**	**0.39**	**0.74**	**0.28**	**0.50**
Qwen2-57B-A14B (64 experts, top-8 activated)											
0%	Original	5.12	9.18	0.33	0.75	0.74	0.63	0.46	0.81	0.39	0.59
40%	NAEE (2024)	8.79	12.64	0.28	0.72	0.71	0.54	0.39	0.76	0.32	0.53
	TD-MoE (2026)	8.07	12.50	0.29	**0.77**	0.71	0.54	0.45	0.78	0.36	0.56
	D^2-MoE (2025)	8.19	11.23	**0.34**	0.75	**0.75**	0.61	0.45	0.79	0.36	0.58
	GS-MoE (Ours)	**6.59**	**9.81**	**0.34**	**0.77**	**0.75**	**0.62**	**0.47**	**0.81**	**0.39**	**0.59**
60%	NAEE (2024)	14.79	23.52	0.21	0.58	0.61	0.44	0.29	0.69	0.26	0.44
	TD-MoE (2026)	12.49	19.92	0.28	0.73	0.68	0.46	0.41	0.74	0.31	0.52
	GS-MoE (Ours)	**7.87**	**11.29**	**0.32**	**0.79**	**0.78**	**0.59**	**0.50**	**0.80**	**0.38**	**0.59**
Mixtral-8×7B-v0.1 (8 experts, top-2 activated)											
0%	Original	3.98	12.99	0.36	0.84	0.76	0.65	0.57	0.82	0.43	0.63
20%	MoE-SVD (2025b)	4.86	19.42	0.33	0.79	0.74	0.56	0.49	0.78	0.37	0.58
	TD-MoE (2026)	**4.49**	17.20	0.33	**0.83**	**0.77**	**0.64**	**0.53**	**0.82**	0.40	**0.62**
	D^2-MoE (2025)	4.65	16.32	0.33	0.80	0.75	0.61	0.51	0.81	0.39	0.60
	GS-MoE (Ours)	5.88	**15.69**	**0.36**	0.82	0.76	0.62	**0.53**	**0.82**	**0.41**	**0.62**
40%	MoE-SVD (2025b)	6.74	27.73	0.27	0.72	0.67	0.43	0.38	0.71	0.32	0.50
	TD-MoE (2026)	5.79	24.60	0.28	0.77	**0.76**	**0.57**	0.47	**0.79**	0.35	**0.57**
	D^2-MoE (2025)	**5.28**	**20.54**	**0.32**	0.78	0.73	**0.57**	0.47	0.78	0.34	**0.57**
	GS-MoE (Ours)	9.29	22.17	0.31	**0.79**	0.70	0.55	**0.49**	**0.79**	**0.38**	**0.57**
60%	MoE-SVD (2025b)	13.52	130.26	0.19	0.45	0.55	0.33	0.23	0.62	0.25	0.37
	TD-MoE (2026)	17.78	79.43	0.21	0.58	0.62	0.38	0.28	0.65	0.27	0.43
	D^2-MoE (2025)	**6.46**	**23.63**	**0.28**	**0.72**	**0.71**	**0.51**	0.38	**0.73**	0.31	**0.52**
	GS-MoE (Ours)	15.94	35.96	0.26	**0.72**	0.64	0.44	**0.41**	**0.73**	**0.32**	0.50

4 EXPERIMENTS

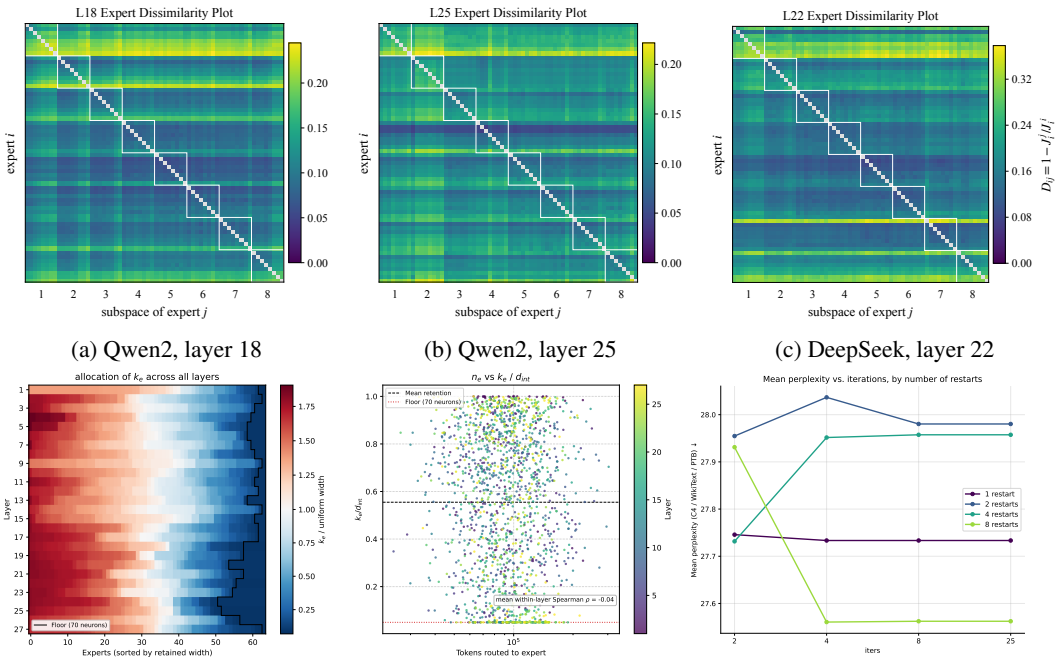
4.1 SETTINGS

Models. We evaluate GS-MoE on Mixtral-8x7B, DeepSeek-MoE-16B, and Qwen2-57B-A14B, covering both coarse-grained and fine-grained MoE architectures. We evaluate compression along shared-subspace rank and intermediate width separately in Table 6 and jointly in Table 1.

Implementations. Compression and evaluation use one to two NVIDIA B200 GPUs, with compression requiring a single GPU. For neuron selection in Section 3.3, we retain at least 5% of each expert's intermediate width and choose $\eta \in \{0.01, 0.1\}$. The balanced medoid clustering in Section 3.4 uses 8 restarts and at most 25 Lloyd iterations, with pairwise gains computed in FP32. For rank allocation in Section 3.5, group ranks range from $\max(\lfloor 0.35 \cdot r_\ell \rfloor, 8)$ to $\min(\lfloor 2.5 \cdot r_\ell \rfloor, d)$. GS-MoE requires no post-compression training, and we report achieved compression after rounding and discrete allocation. Additional numerical and allocation settings are provided in Appendix B. Reported compression rates measure the reduction in total model parameters relative to the uncompressed model.

Table 2: Ablation of grouped assignment and group rank allocation for GS-MoE factorization only on DeepSeek-MoE-16B-Base at 60% compression with $G = 8$. Lower perplexity is better.

Grouped assignment	Group rank allocation	Wiki.	C4	PTB
✗	✗	29.49	33.64	36.49
✗	✓	29.42	33.89	36.56
✓	✗	25.23	28.83	30.35
✓	✓	**25.02**	**28.30**	**29.37**



(a) Qwen2, layer 18 (b) Qwen2, layer 25 (c) DeepSeek, layer 22

(d) Width allocation per layer (e) Width vs. routing load (f) Expert-subspace grouping

Figure 2: (a–c) Expert-to-expert dissimilarity via Obj. 7 on Qwen2-57B-A14B and DeepSeek-MoE-16B-Base. White boxes indicate assigned groups; darker colors indicate lower dissimilarity. (d) Leverage-based retained width k_e relative to a uniform split within each layer on DeepSeek-MoE-16B-Base at 60% compression ($\nu = 0.55$), with experts sorted by width; the black line marks the 70-neuron floor. (e) Under the same setting, k_e/d_{ff} versus tokens routed to each expert (mean within-layer Spearman $\rho = -0.04$). (f) Sensitivity of expert-subspace grouping to the number of restarts and iterations.

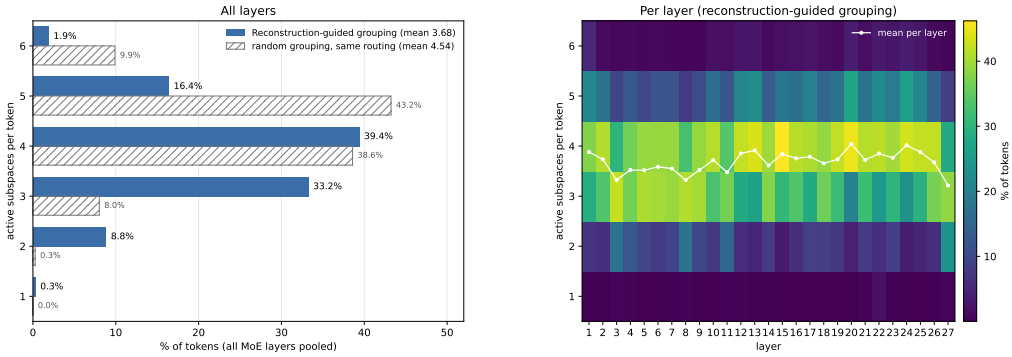
Datasets. We collect calibration statistics in FP64 using a single forward pass over 512 Alpaca samples (Taori et al., 2023), each containing 2048 tokens. We evaluate on nine datasets, including C4 (Raffel et al., 2020) and WikiText-2-raw-v1(Merity et al., 2016) for language modeling, and use the LM Evaluation Harness (Gao et al., 2021) for zero-shot tasks OpenbookQa (Mihaylov et al., 2018), HellaSwag (Zellers et al., 2019), ARC-c (Clark et al., 2018), ARC-e, PIQA (Bisk et al., 2020), Winogrande (bib, 2019), and mathqa (Amini et al., 2019).

Baselines. We compare GS-MoE with MoE-SVD (Li et al., 2025b), D^2-MoE (Gu et al., 2025), TD-MoE (Xu et al., 2026), NAEE (Lu et al., 2024), and MoE-I^2 (Yang et al., 2024).

4.2 Results

Zero-shot performance. We evaluated GS-MoE on seven zero-shot reasoning tasks and compared it to leading MoE compression baselines. Table 1 demonstrates that GS-MoE achieves superior accuracy at higher compression ratios on Qwen2-57B-A14B and Deepseek-MoE-16B-Base, with Qwen2 attaining lossless accuracy at 60% compression and Deepseek retaining 90% of reasoning accuracy. We compare Deepseek-MoE-16B-Base and Qwen2-57B-A14B to D^2-MoE, which utilizes a single base-weight plus expert-specific residuals, and find that GS-MoE outperforms D^2-MoE significantly at all compression ratios for these models. GS-MoE effectively compresses Mixtral-8x7B at compression ratios up to 40%, matching or beating other leading methods.

Sensitivity to clustering iterations and restarts. We evaluate the number of Lloyd-style iterations (Lloyd, 1982) and independent restarts with k-means++ seeding (Arthur & Vassilvitskii, 2007) in our balanced medoid clustering. As shown in Figure 2f, two iterations with two restarts recover



(a) Active subspaces per token.

(b) Active subspaces across all MoE Layers

Figure 3: Active subspaces per token on DeepSeek-MoE-16B-Base ($G = 8$), counted as distinct groups among the top-6 experts. (a) Guided versus random grouping over all tokens and layers. (b) Per-layer distributions for guided grouping, with means shown in the white line.

98% of the largest gain observed across the tested configurations. The lowest perplexity is obtained with four iterations and eight restarts; further iterations slightly increase perplexity.

Grouped assignment and subspace rank allocation. Table 2 shows that grouped assignment reduces WikiText perplexity from 29.49 to 25.23. Adding group rank allocation further improves perplexity across all three datasets, with an average reduction of 0.57. Without grouped assignment, rank allocation provides no consistent improvement.

Table 3: Throughput (tokens/sec, ↑) at 60% compression.

Method	Tokens/sec
Mixtral-8x7B	
D^2-MoE (Gu et al., 2025)	112.84
GS-MoE ($G = 1$)	349.53
GS-MoE ($G = 2$)	337.21
DeepSeek-MoE-16B-Base	
GS-MoE ($G = 1$)	239.06
GS-MoE ($G = 2$)	210.21
GS-MoE ($G = 4$)	207.42
GS-MoE ($G = 8$)	200.22

Runtime efficiency analysis. Table 3 reports throughput at 60% compression. For each model, we benchmark all methods under identical inference settings with B200 GPU(s). On Mixtral-8x7B, GS-MoE achieves 349.53 tokens/sec with $G = 1$, compared with 112.84 for D^2-MoE; increasing G to 2 reduces throughput by 3.5%. On DeepSeek-MoE-16B-Base, increasing G from 1 to 8 reduces throughput by 16.2%. Figure 3a suggests a possible explanation for this moderate overhead: with $G = 8$, each token activates an average of only 3.68 subspaces. Our grouping reduces this average by approximately 19% relative to random grouping under identical routing.

5 Conclusion

In this work, we introduce GS-MoE, a novel training-free compression framework that addresses MoE parameter overhead through group-wise shared subspaces and intra-expert intermediate width reduction. We propose a novel grouping framework that accurately groups experts into error-minimized shared subspaces. Explicit group assignment drastically improves the efficiency of weight-sharing against fine-grained architectures and yields minimal throughput-overhead relative to accuracy improvement. In addition to our expert-grouping framework, we provide a complementary eigenvalue-based group-rank allocation scheme and guide expert-width allocation through leverage scores. We find that, for fine-grained architectures, combining group-wise subspaces and expert intermediate width reduction into a single compression framework retains state-of-the-art fidelity under aggressive parameter budgets.

AI USE STATEMENT

In this work, we used generative AI tools to improve readability and expand our manual search for relevant literature. We did not use generative AI to design or evaluate research methodologies or experiments, formulate or prove mathematical claims, support qualitative and thematic data analysis, or interpret results. Other tasks such as generating synthetic datasets, developing theoretical models, translating text, cleaning datasets, and proposing hypotheses were not applicable to this study. We rigorously reviewed all AI-assisted work, including code and artifacts. Specifically, we manually verified the correctness of all AI-generated code and independently sourced and checked the relevance and bibliographic entries for all AI-suggested related work.

REFERENCES

Winogrande: An adversarial winograd schema challenge at scale. 2019.

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, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1245. URL https://aclanthology.org/N19-1245.

David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1027–1035, 2007. URL https://theory.stanford.edu/~sergei/papers/kMeansPP-soda.pdf.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. URL https://web.stanford.edu/~boyd/cvxbook/.

Inesh Chakrabarti, Sourjya Roy, Bowen Bao, Thiago Crepaldi, Spandan Tiwari, and Ashish Sirasao. Shape mutating expert compression: LorExperts and BTExperts, 2026. URL https://arxiv.org/abs/2608.07814.

I-Chun Chen, Hsu-Shen Liu, Wei-Fang Sun, Chen-Hao Chao, Yen-Chang Hsu, and Chun-Yi Lee. Retraining-free merging of sparse moe via hierarchical clustering. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=hslOzRxzXL.

Xiaodong Chen, Mingming Ha, Zhenzhong Lan, Jing Zhang, and Jianguo Li. MoBE: Mixture-of-basis-experts for compressing MoE-based LLMs. In *The Fourteenth 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:1803.05457v1*, 2018.

David F. Crouse. On implementing 2d rectangular assignment algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, 52(4):1679–1696, 2016. doi: 10.1109/TAES.2016.140952.

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.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, September 2021. URL https://doi.org/10.5281/zenodo.5371628.

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 *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=ziezViPoN1.

Haoze He, Xinkai Zou, Xuan Jiang, Xingyuan Ding, Ao Qu, Juncheng Billy Li, and Heather Miller. Less is moe: Trimming experts in domain-specialist language models, 2026a. URL https://arxiv.org/abs/2606.05538.

Jiangyang He, Shaolin Zhu, and Deyi Xiong. TENP: Trapezoidal expert neuron pruning for mixture-of-experts. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 20911–20925, San Diego, California, United States, July 2026b. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.1049. URL https://aclanthology.org/2026.findings-acl.1049/.

Mike Lasby, Ivan Lazarevich, Nish Sinnadurai, Sean Lie, Yani Ioannou, and Vithursan Thangarasa. REAP the experts: Why pruning prevails for one-shot moe compression. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=ukGxWd2aDG.

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

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 Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 35209–35230. PMLR, 13–19 Jul 2025b. URL https://proceedings.mlr.press/v267/li25az.html.

Chi-Heng Lin, Shangqian Gao, James Seale Smith, Abhishek Patel, Shikhar Tuli, Yilin Shen, Hongxia Jin, and Yen-Chang Hsu. ModeGPT: Modular decomposition for large language model compression. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=8EfxjTCg2k.

Zehua Liu, Han Wu, Ruifeng She, Xiaojin Fu, Xiongwei Han, Tao Zhong, and Mingxuan Yuan. MoLAE: Mixture of latent experts for parameter-efficient language models, 2025. URL https://arxiv.org/abs/2503.23100.

Stuart P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28 (2):129–137, 1982. doi: 10.1109/TIT.1982.1056489.

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 Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6159–6172, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.334. URL https://aclanthology.org/2024.acl-long.334/.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.

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 *EMNLP*, 2018.

Michael Patriksson. A survey on the continuous nonlinear resource allocation problem. *European Journal of Operational Research*, 185(1):1–46, 2008. doi: 10.1016/j.ejor.2006.12.006.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

Samyam Rajbhandari, Conglong Li, Zhewei Yao, Minjia Zhang, Reza Yazdani Aminabadi, Ammar Ahmad Awan, Jeff Rasley, and Yuxiong He. Deepspeed-moe: Advancing mixture-of-experts inference and training to power next-generation ai scale, 2022. URL https://arxiv.org/abs/2201.05596.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

Christopher K. I. Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems*, volume 13, 2000. URL https://proceedings.neurips.cc/paper/2000/hash/19de10adbaa1b2ee13f77f679fa1483a-Abstract.html.

Yuebin Xu, Yanhong Wang, Xuemei Peng, Hui Zang, Minghao Chen, Pengfei Xia, and Zeyi Wen. TD-moe: Tensor decomposition for moe models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=D9cnZNZfxX.

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 Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 10456–10466, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.612. URL https://aclanthology.org/2024.findings-emnlp.612/.

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*, 2019.

Yixiao Zhou, Ziyu Zhao, Dongzhou Cheng, Zhiliang Wu, Jie Gui, Yi Yang, Fei Wu, Yu Cheng, and Hehe Fan. Dropping experts, recombining neurons: Retraining-free pruning for sparse mixture-of-experts LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 15169–15186. Association for Computational Linguistics, November 2025. doi: 10.18653/v1/2025.findings-emnlp.820. URL https://aclanthology.org/2025.findings-emnlp.820/.

A Method Details and Derivations

We use the notation of Section 3.3 and fix one group $\mathcal{E}_g$, its rank r_g, and the neuron selections S_e. The statistics C_e^x, C_e^h, and C_e^y are computed from the original model. The final expert-specific projections have dimensions $A_e^{\uparrow} \in \mathbb{R}^{k_e \times r_g}$ and $A_e^{\downarrow} \in \mathbb{R}^{r_g \times k_e}$; only the temporary down projections $B_e \in \mathbb{R}^{r_g \times d_{\text{ff}}}$ retain the original width. We write $\text{eig}_r(H)$ for the leading r orthonormal eigenvectors of a symmetric matrix, and $\text{orth}(M)$ for an orthonormal basis of its column space. Inverse expressions below assume that the relevant matrices are nonsingular.

A.1 Up Projection Solution

Expert-specific projections for a fixed shared projection. Differentiating the up reconstruction error with respect to $A_e^{\uparrow}$ yields the normal equations $A_e^{\uparrow}((P_g^{\uparrow})^{\top} C_e^x P_g^{\uparrow}) = S_e^{\top} W_e^{\uparrow} C_e^x P_g^{\uparrow}$. Consequently,

$$A_e^{\uparrow} = S_e^{\top} W_e^{\uparrow} C_e^x P_g^{\uparrow} \left((P_g^{\uparrow})^{\top} C_e^x P_g^{\uparrow}\right)^{-1}. \tag{12}$$

The least-squares problem is convex in the expert-specific projections, so this solution minimizes each expert's error for fixed $P_g^{\uparrow}$. With singular projected statistics, the pseudoinverse gives a minimizing solution instead.

Why the shared projection requires an approximation. Substituting Eq. (12) into Obj. 2 leaves a projection-independent term minus

$$\sum_{e \in \mathcal{E}_g} \text{tr}\left[\left((P_g^{\uparrow})^{\top} C_e^x P_g^{\uparrow}\right)^{-1} (P_g^{\uparrow})^{\top} C_e^x (W_e^{\uparrow})^{\top} S_e S_e^{\top} W_e^{\uparrow} C_e^x P_g^{\uparrow}\right]. \tag{13}$$

The inverse term depends on both the expert and the shared projection. For general heterogeneous C_e^x, this expression does not reduce to a single fixed-matrix eigenproblem. We therefore use the group-level approximation below to estimate the shared projection.

Group-level surrogate. Let $\tau_e = n_e / \sum_{j \in \mathcal{E}_g} n_j$, where n_e counts routed calibration assignments, and set $C_g = \sum_{e \in \mathcal{E}_g} \tau_e C_e^x$. Replacing the expert input statistics by C_g and weighting expert contributions by τ_e gives the surrogate

$$\min_{P_g^{\uparrow}, \{A_e^{\uparrow}\}} \sum_{e \in \mathcal{E}_g} \tau_e \left\| (S_e^{\top} W_e^{\uparrow} - A_e^{\uparrow} (P_g^{\uparrow})^{\top}) C_g^{1/2} \right\|_F^2, \qquad (P_g^{\uparrow})^{\top} P_g^{\uparrow} = I_{r_g}. \tag{14}$$

Assume $C_g \succ 0$. For this derivation only, let Q be an orthonormal basis of the column space of $C_g^{1/2} P_g^{\uparrow}$. After minimizing over the expert-specific projections, the surrogate becomes $\text{tr}(H_g^{\uparrow}) - \text{tr}(Q^{\top} H_g^{\uparrow} Q)$, where

$$H_g^{\uparrow} = C_g^{1/2} \left(\sum_{e \in \mathcal{E}_g} \tau_e (W_e^{\uparrow})^{\top} S_e S_e^{\top} W_e^{\uparrow} \right) C_g^{1/2}. \tag{15}$$

Its minimum is the eigenvalue tail $\sum_{i > r_g} \lambda_i(H_g^{\uparrow})$, attained by the leading eigenspace. Mapping this space back gives the shared projection in Eq. (3). Orthonormalization changes the coordinates but not the represented subspace. We then refit the expert-specific projections with the original C_e^x using Eq. (12). This projection is optimal for the surrogate, not necessarily for the original up objective. The same derivation applies to the gate projection with the same selected neurons.

A.2 Down Projection Solution

Eliminating the temporary projections. For any orthonormal $P_g^{\downarrow}$, decompose the output error into orthogonal components:

$$\begin{aligned}\mathbb{E}_e\|y_e - P_g^{\downarrow} B_e h_e\|_2^2 = \mathbb{E}_e\|(I - P_g^{\downarrow}(P_g^{\downarrow})^\top) y_e\|_2^2 \\ + \mathbb{E}_e\|(P_g^{\downarrow})^\top y_e - B_e h_e\|_2^2.\end{aligned} \tag{16}$$

Since $y_e = W_e^{\downarrow} h_e$, choosing $B_e = (P_g^{\downarrow})^\top W_e^{\downarrow}$ makes the second term zero. This choice of B_e globally minimizes the error for the fixed shared projection even when C_e^h is singular; uniqueness is not required.

Optimal shared projection. The remaining unweighted objective is $\sum_e \mathrm{tr}(C_e^y) - \mathrm{tr}((P_g^{\downarrow})^\top (\sum_e C_e^y) P_g^{\downarrow})$. Hence a global solution to Obj. 4 is

$$P_g^{\downarrow} = \mathrm{eig}_{r_g}\left(\sum_{e \in \mathcal{E}_g} C_e^y\right), \qquad B_e = (P_g^{\downarrow})^\top W_e^{\downarrow}. \tag{17}$$

The minimum reconstruction error is $\sum_{i > r_g} \lambda_i(\sum_{e \in \mathcal{E}_g} C_e^y)$. Thus the shared and temporary expert-specific projections can be solved jointly by an eigendecomposition.

For frequency-weighted fitting, multiply each expert term in the objective by τ_e. The same derivation then gives $P_g^{\downarrow} = \mathrm{eig}_{r_g}(\sum_{e \in \mathcal{E}_g} \tau_e C_e^y)$, with the same formula for B_e. This solves the weighted objective, which need not minimize the displayed unweighted objective. Accordingly, $H_g^{\downarrow}$ in Section 3.5 uses the unweighted sum to match Obj. 4.

A.3 Down Projections after Intermediate-Width Compression

With B_e fixed, the retained coordinates are $S_e^\top h_e$. The final expert-specific projection solves

$$\min_{A_e^{\downarrow}} \mathbb{E}_e\|B_e h_e - A_e^{\downarrow} S_e^\top h_e\|_2^2. \tag{18}$$

Its normal equations are $A_e^{\downarrow}(S_e^\top C_e^h S_e) = B_e C_e^h S_e$, giving the Nyström reconstruction in Eq. (6): $A_e^{\downarrow} = B_e C_e^h S_e (S_e^\top C_e^h S_e)^\dagger$. This is a least-squares minimizer for fixed B_e and S_e under the original hidden activations. It does not jointly optimize neuron selection or account for changes to hidden activations caused by the compressed up/gate projections. Therefore, the exact down solution before width compression does not imply global optimality of the final compressed expert.

A.4 Derivation of the Grouping Objective

Fix the candidate shared projections, group ranks, and neuron selections. Before eliminating the expert-specific projections, assignment minimizes the sum of the up and down reconstruction errors:

$$\begin{aligned}\min_{g(\cdot), \{A_e^{\uparrow}, B_e\}} \quad \sum_{e=1}^{E} \mathbb{E}_e\Big[\|S_e^\top W_e^{\uparrow} x - A_e^{\uparrow}(P_{g(e)}^{\uparrow})^\top x\|_2^2 \\ + \|y_e - P_{g(e)}^{\downarrow} B_e h_e\|_2^2\Big].\end{aligned} \tag{19}$$

The assignment satisfies the same group-size constraints as Obj. 7. For the derivation below, consider assigning expert e to a fixed candidate group g. Its optimal projections have dimensions $A_e^{\uparrow} \in \mathbb{R}^{k_e \times r_g}$ and $B_e \in \mathbb{R}^{r_g \times d_{\mathrm{ff}}}$.

Squared-error expansion. For fixed candidate projections, let $A_e^{\uparrow}$ and B_e take their least-squares values for each candidate assignment. Writing $g = g(e)$ within each summand, the assignment-

independent terms can be removed:

$$
\begin{aligned}
&\arg\min_{g(\cdot)} \sum_e \mathbb{E}_e\big[\|S_e^\top W_e^\uparrow x - A_e^\uparrow (P_g^\uparrow)^\top x\|_2^2 + \|y_e - P_g^\downarrow B_e h_e\|_2^2\big] \\
= \arg\min_{g(\cdot)} &\Bigg\{ \underbrace{\sum_e \mathbb{E}_e\big[\|S_e^\top W_e^\uparrow x\|_2^2 + \|y_e\|_2^2\big]}_{\text{independent of assignment}} \\
&\quad + \underbrace{\sum_e \mathbb{E}_e \begin{bmatrix} \|A_e^\uparrow (P_g^\uparrow)^\top x\|_2^2 + \|P_g^\downarrow B_e h_e\|_2^2 \\ -\, 2\langle S_e^\top W_e^\uparrow x, A_e^\uparrow (P_g^\uparrow)^\top x\rangle - 2\langle y_e, P_g^\downarrow B_e h_e\rangle \end{bmatrix}}_{\text{dependent on assignment}} \Bigg\} \\
= \arg\max_{g(\cdot)} &\sum_e \underbrace{\mathbb{E}_e \begin{bmatrix} 2\langle S_e^\top W_e^\uparrow x, A_e^\uparrow (P_g^\uparrow)^\top x\rangle + 2\langle y_e, P_g^\downarrow B_e h_e\rangle \\ -\, \|A_e^\uparrow (P_g^\uparrow)^\top x\|_2^2 - \|P_g^\downarrow B_e h_e\|_2^2 \end{bmatrix}}_{J_e^g}.
\end{aligned} \tag{20}
$$

The remaining per-expert score is J_e^g; the following steps evaluate it using the calibration second moments.

Up reconstruction error. Expanding the expectation gives

$$
\begin{aligned}
&\mathbb{E}_e\|S_e^\top W_e^\uparrow x - A_e^\uparrow (P_g^\uparrow)^\top x\|_2^2 \\
&\quad = \operatorname{tr}\big[S_e^\top W_e^\uparrow C_e^x (W_e^\uparrow)^\top S_e\big] - 2\operatorname{tr}\big[A_e^\uparrow (P_g^\uparrow)^\top C_e^x (W_e^\uparrow)^\top S_e\big] \\
&\qquad + \operatorname{tr}\big[A_e^\uparrow (P_g^\uparrow)^\top C_e^x P_g^\uparrow (A_e^\uparrow)^\top\big].
\end{aligned} \tag{21}
$$

At the least-squares solution in Eq. (12), the normal equations make the final trace equal to the cross trace. Substitution therefore gives

$$
\begin{aligned}
&\min_{A_e^\uparrow} \mathbb{E}_e\|S_e^\top W_e^\uparrow x - A_e^\uparrow (P_g^\uparrow)^\top x\|_2^2 \\
&\quad = \operatorname{tr}\big[S_e^\top W_e^\uparrow C_e^x (W_e^\uparrow)^\top S_e\big] \\
&\qquad - \operatorname{tr}\Big[\big((P_g^\uparrow)^\top C_e^x P_g^\uparrow\big)^{-1} (P_g^\uparrow)^\top C_e^x (W_e^\uparrow)^\top S_e S_e^\top W_e^\uparrow C_e^x P_g^\uparrow\Big].
\end{aligned} \tag{22}
$$

The first term depends only on the original expert and selected neurons; the second depends on the candidate shared projection.

Down reconstruction error. The orthogonal decomposition in Eq. (16) and the solution $B_e = (P_g^\downarrow)^\top W_e^\downarrow$ give

$$
\min_{B_e} \mathbb{E}_e\|y_e - P_g^\downarrow B_e h_e\|_2^2 = \operatorname{tr}(C_e^y) - \operatorname{tr}\big[(P_g^\downarrow)^\top C_e^y P_g^\downarrow\big]. \tag{23}
$$

Again, the first term is independent of the candidate group.

Assignment after elimination. Summing the two minimized errors over experts gives $\xi - \sum_e J_e^{g(e)}$, where

$$
\begin{aligned}
\xi &= \sum_e \big\{\operatorname{tr}\big[S_e^\top W_e^\uparrow C_e^x (W_e^\uparrow)^\top S_e\big] + \operatorname{tr}(C_e^y)\big\}, \\
J_e^g &= \operatorname{tr}\Big[\big((P_g^\uparrow)^\top C_e^x P_g^\uparrow\big)^{-1} (P_g^\uparrow)^\top C_e^x (W_e^\uparrow)^\top \\
&\qquad \cdot S_e S_e^\top W_e^\uparrow C_e^x P_g^\uparrow\Big] + \operatorname{tr}\big[(P_g^\downarrow)^\top C_e^y P_g^\downarrow\big].
\end{aligned} \tag{24}
$$

The constant ξ is independent of the assignment, so minimizing the summed error has the same assignments as maximizing $\sum_e J_e^{g(e)}$. Expanding the squared norms directly gives the four-term expression in Obj. 20; evaluating it at the optimal expert-specific projections yields Obj. 7. Thus, the minimizers of Obj. 19 over assignments coincide with those in Obj. 7 after the expert-specific

projections are fitted for each candidate group. For gated experts, the same calculation adds the gate reconstruction error of a zero prediction to ξ and its error reduction to J_e^g. The equivalence assumes fixed candidate projections and selections and does not guarantee improvement after surrogate refitting. For the balanced grouping score in Section 3.4, multiplying the down reconstruction error by the fixed β_{scale} multiplies both its assignment-independent term and its error reduction by the same factor; the elimination and assignment equivalence remain valid for this weighted objective.

A.5 Discrete Group-Rank Allocation

For fixed groups and neuron selections, Obj. 10 has separable scores V_g and per-rank costs c_g. Searching a multiplier for a shared resource constraint is a standard approach in continuous resource allocation (Patriksson, 2008); water-filling provides a related continuous example (Boyd & Vandenberghe, 2004, Example 5.2). These references motivate the multiplier search, rather than establish optimality of the discrete procedure below.

For a multiplier $\lambda \geq 0$, each group independently chooses an integer rank maximizing $V_g(t) - \lambda \cdot c_g \cdot t$ over $r_{\min} \leq t \leq r_{\max}$. Because the eigenvalues defining V_g are non-increasing, successive rank increments have non-increasing gains. Choosing the smaller rank in a tie makes total parameter cost non-increasing in λ. Bisection increases λ when the total cost exceeds $r \cdot \sum_g c_g$ and decreases it when the cost is below that budget, retaining a feasible allocation. This assumes that the minimum allowed ranks fit within the budget.

Starting from a feasible allocation, residual budget can be assigned by repeatedly selecting the affordable next rank increment with the largest gain per parameter, $[V_g(r_g + 1) - V_g(r_g)]/c_g$, subject to the rank upper bounds. The procedure stops when no positive-gain increment fits. Different c_g and integer ranks can leave budget slack, and this multiplier search with greedy completion does not generally guarantee a global optimum of Obj. 10.

A.6 Layer Allocation and Budget Correction

This section details the across-layer allocation in Section 3.5: selecting the overall rank–width split, distributing compression by layer importance, and correcting the total parameter budget.

Layer budgets and shared overhead. For layers with the same n, E, d, d_{ff}, and G, an uncompressed layer holds $n \cdot E \cdot d \cdot d_{\text{ff}}$ routed parameters. At reference rank r_ℓ, the shared projections hold $n \cdot G \cdot d \cdot r_\ell$ parameters and the expert-specific projections hold $n \cdot r_\ell \cdot K_\ell$. The reference retained fraction is therefore $\mathcal{P}_\ell = \alpha_\ell \cdot (\beta_\ell + c_{\text{shared}})$, where $\alpha_\ell = r_\ell/d$, $\beta_\ell = K_\ell/(E \cdot d_{\text{ff}})$, and $c_{\text{shared}} = G \cdot d/(E \cdot d_{\text{ff}})$, with rank retention α_ℓ, width retention β_ℓ, and overhead c_{shared} of the shared bases. We allocate in terms of $(\alpha_\ell, \beta_\ell)$ and recover integer budgets by rounding and clipping $r_\ell = \lfloor \alpha_\ell \cdot d \rfloor$ and $K_\ell = \lfloor \beta_\ell \cdot E \cdot d_{\text{ff}} \rfloor$ to the prescribed rank and width bounds. Here r_ℓ is the reference rank of Obj. 10 and $K_\ell = \sum_e k_e$. Since $\sum_g c_g = G \cdot d + K_\ell$, Obj. 10 bounds the actual layer parameter count by this reference budget; discrete allocation can leave unused slack.

Global split. Let ρ be the target reduction in total model parameters, and let N and N_{routed} denote the original total and routed-expert parameter counts, respectively. Since parameters outside the routed experts remain unchanged, the routed-expert budget is $N_{\text{routed}} - \rho \cdot N$, giving the retained fraction $R = 1 - \rho \cdot N/N_{\text{routed}}$. We divide this routed-expert retention between width and rank with an exponent $\nu \in [0, 1)$: $\bar{\beta} = R^\nu$ and $\bar{\alpha} = R/(\bar{\beta} + c_{\text{shared}})$, so that $\bar{\alpha}(\bar{\beta} + c_{\text{shared}}) = R$ under uniform allocation, with $\nu = 0$ recovering pure factorization ($\bar{\beta} = 1$). Pure width pruning is a separate configuration in which the shared bases are dropped altogether ($\alpha_\ell \equiv 1$, $c_{\text{shared}} = 0$, $\bar{\beta} = R$); it is not the $\nu \to 1$ limit of this formula.

For L layers with positive importance scores $I = (I_1, \ldots, I_L)$, we distribute width and rank removal using $p^{(a)} = \text{softmax}(-\log_{10} I/\epsilon_a)$ for $a \in \{w, r\}$, with temperature $\epsilon_a > 0$. Clipping the per-layer removals $L \cdot (1 - \bar{\beta}) \cdot p_\ell^{(w)}$ and $L \cdot (1 - \bar{\alpha}) \cdot p_\ell^{(r)}$ and redistributing excess removal gives preliminary retentions $\hat{\beta}_\ell$ and $\hat{\alpha}_\ell$. We keep $\beta_\ell = \hat{\beta}_\ell$ and rescale ranks as $\alpha_\ell = \kappa \cdot \hat{\alpha}_\ell$, where $\kappa = R/[\frac{1}{L} \sum_m \hat{\alpha}_m \cdot (\hat{\beta}_m + c_{\text{shared}})]$, to obtain $\frac{1}{L} \sum_\ell \mathcal{P}_\ell = R$ in the continuous relaxation, provided the preliminary allocations and rescaled ranks are feasible. Rounding and discrete allocation can change the achieved compression.

Table 4: Default compression hyperparameters for the main comparisons. Ablation settings are specified separately.

Model	Compression Rates	η	ν	G
DeepSeek	20%	0.1	0.75	1
	40%	0.1	0.65	2
	60%	0.1	0.55	8
Mixtral	20%	0.01	0.85	1
	40%	0.01	0.85	1
	60%	0.01	0.85	1
Qwen	20%	0.1	0.75	1
	40%	0.1	0.65	2
	60%	0.1	0.55	8

For DeepSeek and Qwen with 256 calibration samples, $\eta = 0.01$ was more effective; the main comparisons with 512 calibration samples use $\eta = 0.1$ for these models.

B Additional Experimental Settings

We give the numerical and allocation details for the experiments in Section 4.1. Neuron selection and projection fitting follow Section 3.3, expert grouping follows Section 3.4, and capacity allocation follows Section 3.5.

Neuron selection. We retain at least 5% of the original intermediate width per expert, with the layer width floor set to $E \cdot \lfloor 0.05 \cdot d_{\text{ff}} \rfloor$. The ridge parameter is selected from $\eta \in \{0.01, 0.1\}$ according to model type and calibration size.

Unless otherwise specified, Mixtral-8x7B uses $\eta = 0.01$, while DeepSeek-MoE-16B-Base and Qwen2-57B-A14B use $\eta = 0.1$, with 512 Alpaca calibration samples of 2048 tokens each.

Experts with few routed calibration inputs. For experts with fewer than $2 \cdot d_{\text{ff}}$ hidden-activation or hidden-state samples, we feed randomly sampled hidden-states and activation batches until at least $2 \cdot d_{\text{ff}}$ samples are collected. Experts with fewer than $2 \cdot d_{\text{ff}}$ routed input tokens are excluded from the shared update and use the fallback $A_e^{\uparrow} = S_e^{\top} W_e^{\uparrow} C_g P_g^{\uparrow} ((P_g^{\uparrow})^{\top} C_g P_g^{\uparrow})^{-1}$. These experts receive widths close to the minimum because their aggregate neuron leverage scores are smaller. Qwen2-57B-A14B contains experts with zero routed calibration tokens; the other evaluated models have sufficiently routed experts.

Input-statistics regularization. For numerical computation, we use $C_g = \sum_{e \in \mathcal{E}_g} \tau_e (C_e^x + \eta_x \cdot \text{tr}(C_e^x) \cdot I)$ with $\eta_x = 1.0$. This regularized matrix replaces C_g in the up surrogate construction.

Grouping. Balanced medoid clustering uses k-means++ seeding with 8 restarts and at most 25 Lloyd iterations. The pairwise gain sweep is computed in FP32.

Rank and layer allocation. Group ranks satisfy $r_{\min} = \max(\lfloor 0.35 \cdot r_\ell \rfloor, 8)$ and $r_{\max} = \min(\lfloor 2.5 \cdot r_\ell \rfloor, d)$. Layer reference ranks are recovered as $r_\ell = \text{clip}(\lfloor \alpha_\ell \cdot d \rfloor, 8, d)$, and total widths as $K_\ell = \text{clip}(\lfloor \beta_\ell \cdot E \cdot d_{\text{ff}} \rfloor, E \cdot \lfloor 0.05 \cdot d_{\text{ff}} \rfloor, E \cdot d_{\text{ff}})$. For either allocation axis, layer removal fractions are clipped to $[0.02, \omega]$, where $\omega = 0.85$, increasing to 0.95 when the corresponding aggregate removal exceeds 0.40. Clipped mass is redistributed over layers that have not reached their bounds. We report achieved compression after integer rounding and discrete rank allocation.

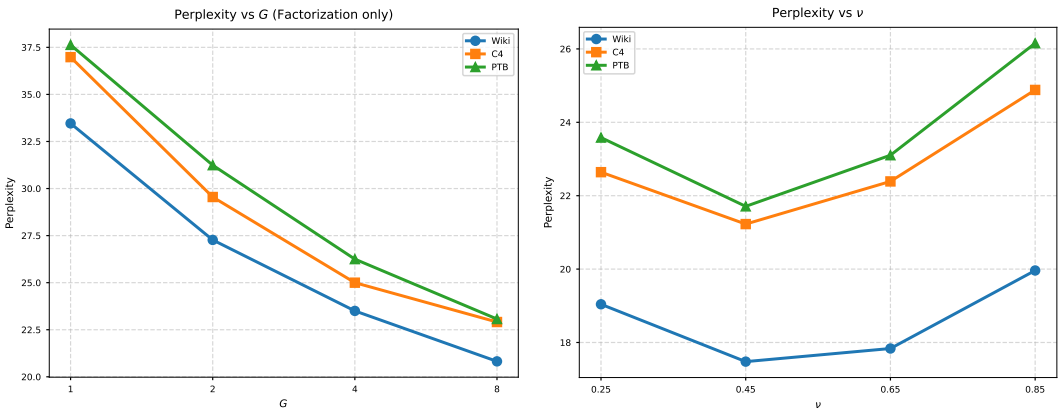
C Additional Experiments

Effect of G and ν. Figure 4 (left) varies the number of groups with factorization leaving intermediate width full-rank. Perplexity decreases monotonically with G on all three datasets, with WikiText falling from 33.5 at $G = 1$ to 20.75 at $G = 8$. Returns diminish with each doubling

Table 5: Perplexity (↓) of GS-MoE on Qwen2-57B-A14B under uniform and non-uniform allocation. **(a)** Uniform vs. leverage-based expert widths; ν controls the split between width and rank compression. **(b)** Uniform vs. non-uniform layer-wise allocation, calibrated on Alpaca. Lower perplexity for each dataset and setting is in **bold**.

(a) Per-expert width allocation							
		Uniform			Non-uniform		
Ratio	ν	Wiki	C4	PTB	Wiki	C4	PTB
20%	0.75	6.17	9.08	9.33	**6.07**	**8.98**	**9.26**
40%	0.65	6.83	10.16	10.05	**6.59**	**10.02**	**9.81**
60%	0.55	8.10	12.15	11.65	**7.87**	**11.93**	**11.29**
(b) Layer-wise allocation							
		Uniform			Non-uniform		
Ratio	ν	Wiki	C4	PTB	Wiki	C4	PTB
60%	0.55	8.12	12.34	11.56	**7.87**	**11.93**	**11.29**

(6.1, 3.8, and 2.7 points on WikiText). This is consistent with outlier-experts being more capable of representing features under smaller group sizes. Figure 4 (right) varies width retention by the hyperparameter ν detailed in A.6. Minimum perplexity is found at $\nu = 0.45$, with additional increments increasing perplexity. Smaller ν values restrict expert representation, degrading per-expert resolution, while large values of ν allocate more subspace rank without the proper compensatory compression rate to maximize accuracy.



(a) Effect of $G \in \{1, 2, 4, 8\}$.

(b) Effect of ν at four broadly spaced values.

Figure 4: Perplexity of GS-MoE on DeepSeek-16B-Base at 55% compression. **(a)** Varying the number of groups G. **(b)** Varying ν.

C.1 Non-Uniform Layer and Intermediate Width Allocation

We compare non-uniform and uniform expert width and layer allocation in Table 5. Table 5 demonstrates improvement under non-uniform per-expert intermediate width allocation for Qwen models, improving Wikitext perplexity by 0.23 at 60% compression. Similarly, non-uniform layer-wise allocation reduces Wikitext perplexity by 0.26 at 60% compression.

C.2 Evaluations of GS-MoE's Subsets

Table 6 provides further insight on GS-MoE's performance against fine-grained and coarse-grained architectures. Table 6 lists three competing methods: pure factorization (Fact), pure width reduction (Prune), and the full pipeline (GS-MoE). For Qwen models, pure factorization performs in-line

Table 6: Subsets of GS-MoE on three different models. Factorization is GS-MoE's group shared subspace method with full intermediate width. Nyström is GS-MoE with only expert intermediate width reduction.

Ratio	Method	Wiki.↓	PTB↓	OpenB.	ARC-e	WinoG.	HellaS.	ARC-c	PIQA	MathQA	Avg.↑
	DeepSeekMoE-16B-Base (64 experts, top-6 activated)										
0%	Original	6.38	9.47	0.32	0.75	0.70	0.58	0.44	0.79	0.31	0.56
20%	Factorization	8.11	11.47	0.30	0.74	**0.70**	0.54	0.43	0.77	0.28	0.538
	Nyström	**7.30**	**10.53**	**0.33**	**0.76**	**0.70**	**0.57**	**0.44**	**0.79**	**0.32**	**0.558**
	GS-MoE (Ours)	7.35	10.64	**0.33**	0.75	**0.70**	**0.57**	0.43	**0.79**	0.30	0.552
40%	Factorization	12.97	16.13	0.27	0.71	0.68	0.48	0.39	0.74	0.26	0.506
	Nyström	**10.09**	14.30	0.30	**0.75**	0.69	**0.53**	0.42	**0.78**	**0.31**	0.539
	GS-MoE (Ours)	10.10	**13.51**	**0.31**	**0.75**	**0.70**	**0.53**	**0.43**	**0.78**	0.29	**0.543**
60%	Factorization	25.02	29.37	0.24	0.66	0.64	0.39	0.35	0.70	0.25	0.462
	Nyström	18.62	27.12	0.27	0.71	0.62	0.41	0.37	0.73	**0.29**	0.487
	GS-MoE (Ours)	**17.55**	**22.21**	**0.28**	**0.72**	**0.66**	**0.45**	**0.39**	**0.74**	0.28	**0.503**
	Qwen2-57B-A14B (64 experts, top-8 activated)										
0%	Original	5.12	9.18	0.33	0.75	0.74	0.63	0.46	0.81	0.39	0.59
40%	Factorization	7.19	10.46	0.31	**0.78**	**0.76**	0.59	**0.49**	0.80	**0.40**	0.591
	Nyström	6.61	9.83	0.33	**0.78**	0.75	**0.62**	0.48	**0.81**	0.39	**0.593**
	GS-MoE (Ours)	**6.59**	**9.81**	**0.34**	0.77	0.75	**0.62**	0.47	**0.81**	0.39	0.592
60%	Factorization	9.61	14.47	0.29	0.77	0.75	0.54	0.47	0.79	0.36	0.567
	Nyström	7.89	11.47	**0.32**	0.78	0.73	**0.60**	0.49	**0.80**	**0.38**	0.586
	GS-MoE (Ours)	**7.87**	**11.29**	**0.32**	**0.79**	**0.78**	0.59	**0.50**	**0.80**	**0.38**	**0.594**
	Mixtral-8×7B-v0.1 (8 experts, top-2 activated)										
0%	Original	3.98	12.99	0.36	0.84	0.76	0.65	0.57	0.82	0.43	0.63
20%	Factorization	6.25	**15.64**	0.34	0.81	**0.76**	0.59	0.51	0.81	0.38	0.599
	Nyström	**5.81**	16.05	**0.36**	**0.82**	0.75	**0.62**	**0.53**	**0.82**	**0.41**	**0.618**
	GS-MoE (Ours)	5.88	15.69	**0.36**	**0.82**	**0.76**	**0.62**	**0.53**	**0.82**	**0.41**	0.616
40%	Factorization	20.00	35.08	0.28	0.73	0.70	0.45	0.41	0.74	0.30	0.514
	Nyström	9.56	22.35	**0.31**	**0.79**	**0.71**	**0.55**	0.48	**0.79**	**0.39**	**0.574**
	GS-MoE (Ours)	**9.29**	**22.17**	**0.31**	**0.79**	0.70	**0.55**	**0.49**	**0.79**	0.38	0.572
60%	Factorization	62.10	103.50	0.19	0.51	0.57	0.34	0.27	0.64	0.24	0.394
	Nyström	16.37	37.17	**0.26**	0.70	0.63	0.43	**0.41**	**0.73**	**0.35**	**0.502**
	GS-MoE (Ours)	**15.94**	**35.96**	**0.26**	**0.72**	**0.64**	**0.44**	**0.41**	**0.73**	0.32	**0.502**

with width reduction while at higher compression ratios the two methods alone drop significantly in performance. By incorporating the full-pipeline of GS-MoE, GS-MoE is able to recover the performance loss that occurs at 60% compression of either method. The Deepseek architecture tells a similar story, demonstrating that combining GS-MoE's grouped-subspace factorization with pruning pushes the compression limit higher than what other methods offer. Next, GS-MoE's expert width reduction is effectively able to compress Mixtral-8x7B up to 40% compression, performing better than GS-MoE's factorization approach at the same ratios. However, GS-MoE's full-pipeline struggles to exceed the reasoning and perplexity performance of width-reduction. We attribute this to the significantly larger intermediate dimension of Mixtral-8x7B and reduced expert count, making subspace rank reduction significantly more damaging and limiting the efficacy of GS-MoE's expert subspace assignment.