SLEM: SPECTRUM-AWARE LATENT EXPERT MERGING

Anonymous authors
Paper under double-blind review

ABSTRACT

Mixture-of-Experts (MoE) models scale model capacity efficiently by activating only a small subset of experts for each input, while still requiring all parameters to be stored in memory, resulting in substantial memory overhead. Expert merging techniques has recently emerged as a promising compression paradigm that reduces this overhead by consolidating multiple experts into a smaller set. However, existing approaches have two key limitations: (i) direct convex interpolation in parameter space is not generally guaranteed to yield meaningful interpolation of expert functions; and (ii) estimating expert similarity or merging coefficients often relies on calibration data, introducing dependence on the calibration distribution. To address these limitations, we propose **Spectrum-Aware Latent Expert Merging (SLEM)**, a calibration-free expert merging framework that occurs in a structured latent representation of expert weights using a VAE. SLEM groups experts within and across layers based on latent similarities and derives merging coefficients from expert weight spectra. It then performs geometry-aware latent interpolation and decodes the resulting representations into merged expert weights. Compared with existing expert merging methods, SLEM achieves comparable or better performance on general knowledge benchmarks and higher average performance on generation benchmarks involving mathematical reasoning and code generation.

1 INTRODUCTION

Mixture-of-Experts (MoE) models scale capacity efficiently by activating only a small subset of experts for each token (Shazeer et al., 2017; Cai et al., 2025). This sparsity reduces per-token computation, but not memory usage: all expert parameters must remain stored regardless of how frequently they are activated. As a result, the memory footprint scales with the total number of experts rather than the number used for each token. Expert pruning and expert merging address this mismatch by either removing experts or consolidating multiple experts into a smaller set (He et al., 2023; Li et al., 2024; Miao et al., 2025).

Expert merging is particularly attractive because it seeks to retain information from the experts it consolidates rather than simply discarding them. Existing approaches, however, face two limitations. First, experts are typically combined directly in parameter space, although linear interpolation between neural-network parameters is not generally guaranteed to correspond to meaningful interpolation between the functions they represent. Neuron alignment (Li et al., 2024), curvature-aware merging (Nguyen et al., 2025), and subspace decomposition (Li et al., 2026) mitigate this issue, but the resulting merge remains tied to the original parameter space. Second, deciding which experts to merge and how strongly they should contribute often relies on calibration data through routing frequencies, routing patterns, activations, or expert outputs (Li et al., 2024; Chen et al., 2025; Li et al., 2026). Such statistics make the resulting compression decisions dependent on the calibration distribution. Improving the merging rule alone may therefore be insufficient: both the representation in which experts are combined and the information used to guide compression need to be reconsidered.

These limitations motivate reformulating expert merging as a weight-space learning problem (Han et al., 2026), shifting the merging domain from raw parameter space to a learned latent weight space (Soro et al., 2026) in which MoE experts may admit a geometry better suited to merging without calibration data. Building on this view, we propose **Spectrum-Aware Latent Expert Merging (SLEM)**, a calibration-free framework that merges experts in a structured latent representation learned with a variational autoencoder (VAE) (Higgins et al., 2017). SLEM encodes expert weights, groups experts by latent similarity, aligns subnetworks in the same latent space, performs geometry-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

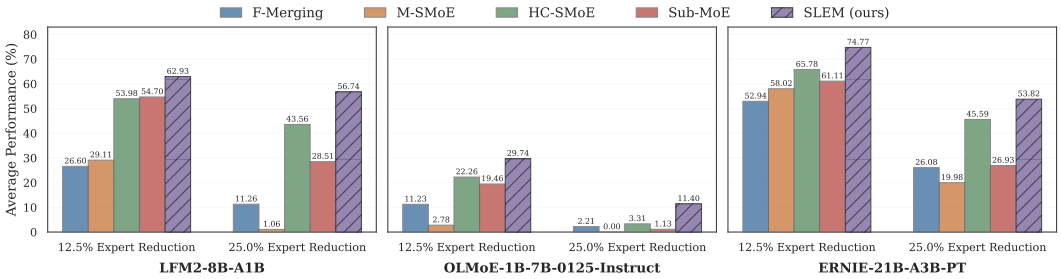


Figure 1: Average performance comparison of expert merging methods on generation-based benchmarks involving mathematical reasoning and code generation (GSM8K, MATH500, MBPP, and HumanEval). SLEM consistently achieves the highest average performance under all evaluated settings.

aware interpolation, and decodes the result into a merged expert, with subnetwork-specific merging coefficients derived from the spectral structure of the pretrained weights. It therefore requires neither calibration data nor additional forward passes: latent geometry replaces direct parameter-space interpolation, while spectral structure replaces input-dependent routing statistics with quantities computed directly from pretrained expert weights.

Our contributions are summarized as follows:

1. We propose **Spectrum-Aware Latent Expert Merging (SLEM)**, which introduces an Expert VAE to learn latent representations of expert weights and performs geometry-aware expert interpolation in the learned latent space.
2. We introduce an expert merging strategy combining spectrum-aware merging coefficients with latent-similarity-based expert grouping, enabling fully calibration-free expert merging without additional data.
3. We extend expert merging beyond within-layer constraints and enable cross-layer expert merging by representing experts from all MoE layers in a shared latent space.
4. We demonstrate that SLEM achieves the highest average performance on mathematical reasoning and code generation benchmarks, with ablations validating latent-space merging, cross-layer merging, and spectrum-aware coefficients.

Taken together, SLEM provides a new perspective on expert merging by changing both interpolation and merging decisions from input-dependent statistics toward the intrinsic structure of expert weights.

2 Related Work

MoE Compression and Expert Merging. One line of work reduces the memory and inference cost of MoE models by pruning redundant parameters or experts (Cai et al., 2025; Liu et al., 2026), scoring them with weight magnitude, activations and router weights (Xie et al., 2024), approximate Shapley values (Huang et al., 2025), learned per-layer ratios (Bai et al., 2025), cross-domain behavior clusters (Hu et al., 2026), or informative tokens (Zong et al., 2026), replacing redundant experts with lightweight novices (Zhang et al., 2026), or exploiting finer-grained parameter structure through low-rank matrix sharing (Li et al., 2025), a shared base with low-rank deltas (Gu et al., 2025), and cross-matrix micro-experts (Xu et al., 2025). Expert merging is complementary, since it preserves information from the experts it consolidates rather than discarding them: MEO approximates several activated experts with one (He et al., 2023), M-SMoE aligns expert neurons and derives groups and coefficients from routing statistics (Li et al., 2024), HC-SMoE clusters experts by output similarity (Chen et al., 2025), Sub-MoE extracts shared subspaces via joint SVD (Li et al., 2026), NAMEx obtains merging weights from a Nash bargaining problem (Nguyen et al., 2026), and PuzzleMoE separates shared from expert-specific parameters for sparse element-wise merging (Zhao et al., 2026). In all of these, the merged expert is still constructed from the original parameters or their explicit decompositions, and grouping and weighting rely on routing frequencies, expert outputs, activations, or calibration examples.

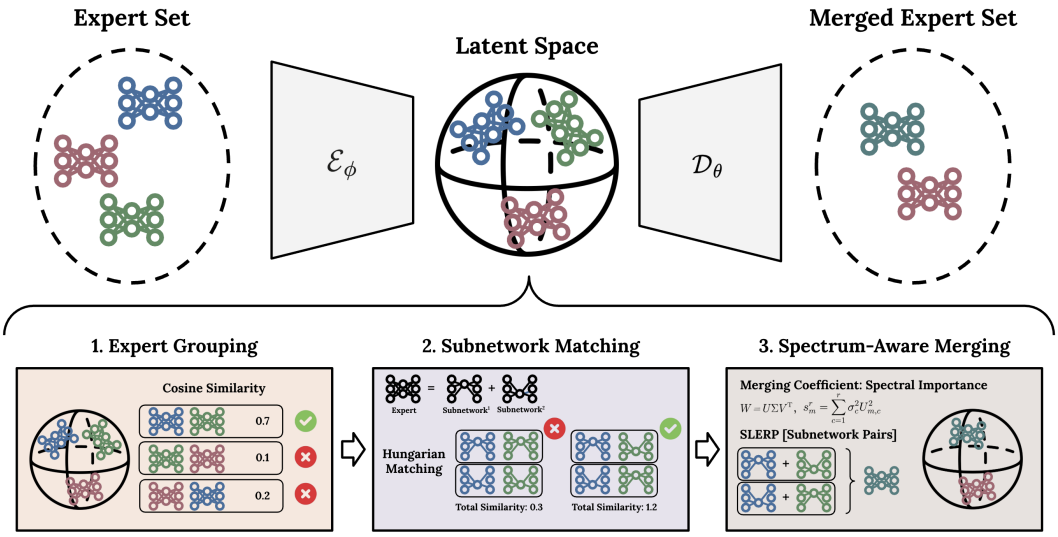


Figure 2: Overall framework of Spectrum-Aware Latent Expert Merging (SLEM).

Weight-Space Learning. A growing line of work treats trained network parameters as a data modality in their own right (Han et al., 2026; Wang et al., 2026). Early efforts learn representations over populations of trained models to predict model characteristics (Schürholt et al., 2021). Later work extends such representations to recurrent weight matrices (Herrmann et al., 2024) and to downstream domains such as geospatial learning, where representations are learned from trained models rather than raw data (Hanna et al., 2026). Because network functions are invariant to neuron permutations, a parallel line designs permutation-equivariant architectures that process weights directly (Zhou et al., 2023a;b), recently extended to the symmetries of Transformer weights (Tran et al., 2025). On the generative side, hyper-representations embed weights from model populations and generate new parameter configurations from them (Schürholt et al., 2022), followed by diffusion-based generators (Soro et al., 2025), flow matching with re-basin alignment to handle permutation symmetry (Gupta et al., 2026), and width-agnostic generation over structurally aligned weight spaces (Kim et al., 2026). Weight-space operations also underlie model editing and merging. Task arithmetic combines fine-tuning deltas directly in parameter space (Ilharco et al., 2023), whereas LS-Merge, closest to our setting, encodes pretrained language-model weights with a Transformer-based variational autoencoder, aligns heterogeneous models in the learned representation, and interpolates them in latent space before decoding back to parameters (Soro et al., 2026). While these methods establish weight space as a useful domain for representation, generation, and model-level manipulation, we study learned weight representations at the granularity of individual MoE experts and use their geometry explicitly for compression. Rather than merging complete models or interpolating experts directly in parameter space, we learn latent representations of individual experts and combine their latent geometry with spectral structure to determine expert groupings and merging coefficients without calibration data.

3 Spectrum-Aware Latent Expert Merging (SLEM)

Consider a pretrained Mixture-of-Experts (MoE) model with L MoE layers, where layer ℓ contains K experts $\{E_1^{(\ell)}, \ldots, E_K^{(\ell)}\}$, of which only $k \ll K$ are activated for each token. Unlike conventional within-layer expert merging, SLEM allows expert pairs to be selected both within and across MoE layers. Accordingly, our objective is to reduce the expert-memory footprint without retraining by replacing an expert pair $\mathcal{G} = \{E_i^{(\ell)}, E_j^{(\ell')}\}$, where ℓ and ℓ' may be identical or different, with a single merged expert. Each pairwise merge therefore reduces the number of stored expert parameter sets by one. Rather than combining the corresponding expert weights directly in weight space, we treat them as structured data, map them to learned latent representations, merge in that space, and decode the result back into a single expert. For notational simplicity, we omit layer indices throughout, while allowing E_i and E_j to originate from either the same or different MoE layers.

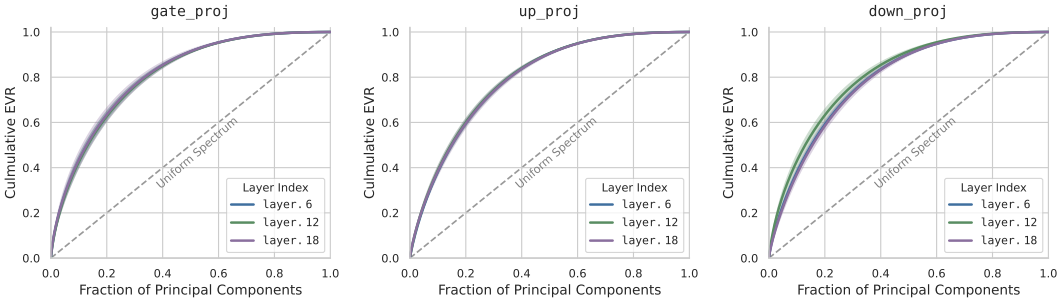


Figure 3: Cumulative explained variance ratio (EVR) of expert projection weights across layers in LFM2-8B-A1B. The EVR curves show that variance is concentrated along a subset of principal directions across all projection types, motivating the use of dominant spectral structure for estimating subnetwork importance. Results for the other evaluated models are provided in Appendix G.

3.1 Hyperspherical Expert VAE

Following LS-Merge (Soro et al., 2026), we learn latent representations of expert weights with transformer-based variational autoencoders (VAEs). To reflect the internal structure of MoE experts, we design an Expert VAE that jointly models the three projection types $p \in \mathcal{P} = \{\texttt{gate}, \texttt{up}, \texttt{down}\}$. The training data consist solely of the non-shared expert weights of the pretrained model, obtained by splitting the fused `gate_up_proj` weights of each expert E_i into gate and up projections and retaining the corresponding `down_proj` weights. Each projection weight is represented as a sequence along the hidden-unit dimension, since a gated feed-forward expert with intermediate width D_{MoE} decomposes into contributions from its hidden units, which we refer to as subnetworks,

$$E_i(x) = \sum_{m=1}^{D_{\text{MoE}}} E_{i,m}(x), \qquad E_{i,m}(x) = \varphi\big(g_{i,m}^\top x\big)\,\big(u_{i,m}^\top x\big)\,d_{i,m}, \tag{1}$$

where $g_{i,m}$ and $u_{i,m}$ are the m-th rows of the gate and up projections respectively, $d_{i,m}$ is the m-th column of the down projection, and φ is the activation function. The triplet $(g_{i,m}, u_{i,m}, d_{i,m})$ defines a subnetwork index common to the three projections. Organizing the projections by this index gives $W_{i,\texttt{gate}} = G_i$, $W_{i,\texttt{up}} = U_i$, and $W_{i,\texttt{down}} = D_i^\top$, all in $\mathbb{R}^{D_{\text{MoE}} \times D}$, where the m-th row of $W_{i,p}$ holds the component of subnetwork m in projection $p \in \mathcal{P}$ and D is the model hidden dimension. A single training sample for each expert is constructed by concatenating the three projection weights along the sequence dimension, $W_i = [W_{i,\texttt{gate}}; W_{i,\texttt{up}}; W_{i,\texttt{down}}] \in \mathbb{R}^{3D_{\text{MoE}} \times D}$. We use type-specific projections to distinguish the gate, up, and down segments while jointly modeling their relationships within each expert. In addition, we omit positional encodings because the absolute ordering of subnetworks has no canonical functional meaning, as reflected in equation 1.

Hyperspherical prior and objective. The prior toward which the posterior is regularized determines the geometry in which expert representations are later compared and interpolated, so its choice is central to our setting. Schürholt et al. (2022) observed that learned weight embeddings tend to occupy a shell in high-dimensional space, which is poorly matched by the Gaussian prior commonly used in VAEs. We therefore adopt a hyperspherical VAE (Davidson et al., 2018) with von Mises–Fisher (vMF) prior and use a fixed concentration κ following prior work (Xu & Durrett, 2018). For an expert weight W, the encoder defines $q_\phi(\mathbf{z} \mid W) = \text{vMF}(\mu_\phi(W), \kappa)$, while the prior is parameterized as $p_\psi(\mathbf{z}) = \text{vMF}(\nu, \kappa)$, where ν is a learnable prior direction. We optimize

$$\mathcal{L} = \min_{\theta,\phi,\psi} \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|W)} \left[\|W - \mathcal{D}_\theta(\mathbf{z})\|_F^2 \right] + \beta D_{\text{KL}}\left(q_\phi(\mathbf{z} \mid W) \,\|\, p_\psi(\mathbf{z})\right), \tag{2}$$

subject to $\|\mathbf{z}\|_2 = \|\mu_\phi(W)\|_2 = \|\nu_\ell\|_2 = 1$, with β controlling the trade-off between reconstruction fidelity and latent regularity (Higgins et al., 2017). Both the sampled latent and the vMF mean directions therefore lie on $\mathbb{S}^{d-1}$; we denote the latent representation of subnetwork m of expert i in projection p as $\mathbf{z}_{i,p}^m$.

3.2 Latent Expert Merging

Given the learned latent space, SLEM (i) identifies expert pairs and aligns their subnetworks using latent similarity, (ii) assigns subnetwork-specific merging coefficients based on spectral importance, and (iii) performs geometry-aware latent interpolation, followed by decoding the merged expert.

Latent subnetwork similarity. Since the latent representations lie on the unit hypersphere, their geometry is characterized by direction, which makes cosine similarity a natural choice. We define the compatibility between subnetwork m of expert i and subnetwork n of expert j as

$$C_{m,n}^{(i,j)} = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \frac{(\mathbf{z}_{i,p}^{m})^{\top} \mathbf{z}_{j,p}^{n}}{\|\mathbf{z}_{i,p}^{m}\|_2 \|\mathbf{z}_{j,p}^{n}\|_2}. \tag{3}$$

Based on equation 3, we define the similarity of the two experts i, j as the mean compatibility over all subnetwork pairs, $\mathcal{S}(i,j) = \frac{1}{D_{\text{MoE}}^2} \sum_{m=1}^{D_{\text{MoE}}} \sum_{n=1}^{D_{\text{MoE}}} C_{m,n}^{(i,j)}$. Expert pairs with higher $\mathcal{S}(i,j)$ are considered more compatible and are therefore prioritized for grouping. We form pairwise merge groups by greedy one-to-one matching on $\mathcal{S}$: at each step we take the most similar remaining pair, assign it to a merge group, and remove both experts from consideration, until the desired number of pairs is reached. We additionally impose a layer-wise limit on the number of merged experts to prevent merge groups from being overly concentrated in a small subset of layers.

Within a group, subnetworks still have no canonical ordering across experts and must be aligned before interpolation. Hungarian matching of hidden units substantially improves expert merging (Li et al., 2024), so we follow this approach but define the criterion in latent space, seeking the permutation π of $[D_{\text{MoE}}]$ that maximizes total compatibility defined as $\pi^{\star} = \arg\max_{\pi} \sum_{m=1}^{D_{\text{MoE}}} C_{m,\pi(m)}^{(i,j)}$. The resulting permutation is applied consistently across the three projections, preserving the shared subnetwork index of equation 1.

Spectral importance as merging coefficients. As shown in Figure 3, subnetworks do not contribute equally to the dominant structure of an expert, so we set their coefficients from the spectrum of the pretrained weights $W_{i,p} = A_{i,p} \Sigma_{i,p} B_{i,p}^{\top}$, where $\sigma_{i,p,c}$ is the c-th singular value and $A_{i,p,mc}$ the (m,c) entry of $A_{i,p}$. We quantify the spectral importance of subnetwork m as

$$\tilde{s}_{i,p,m}^{r} = \sum_{c=1}^{r} \sigma_{i,p,c}^{2} A_{i,p,mc}^{2}, \quad s_{i,p,m}^{r} = \frac{\tilde{s}_{i,p,m}^{r}}{\sum_{n} \tilde{s}_{i,p,n}^{r}}, \quad s_{i,m}^{r} = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} s_{i,p,m}^{r}, \tag{4}$$

which measures the amount of spectral energy in the leading r singular directions attributable to subnetwork m. A larger value therefore indicates that the subnetwork contributes more strongly to the dominant structure of the expert. For a matched pair $(m, \pi^{\star}(m))$, we then determine the merging coefficient from their relative spectral importance, assigning a larger contribution to the more spectrally important subnetwork:

$$\alpha_m = \frac{s_{i,m}^{r}}{s_{i,m}^{r} + s_{j,\pi^{\star}(m)}^{r}}, \quad \alpha_{\pi^{\star}(m)} = \frac{s_{j,\pi^{\star}(m)}^{r}}{s_{i,m}^{r} + s_{j,\pi^{\star}(m)}^{r}} = 1 - \alpha_m. \tag{5}$$

These coefficients replace uniform averaging and routing-frequency weighting, and the rank r, which controls how much of the leading spectrum is retained, is the only free parameter of the rule.

Latent expert interpolation. Because the representations are directional, we merge each aligned pair with spherical linear interpolation as $\widetilde{\mathbf{z}}_{ij}^{m} = \text{SLERP}(\mathbf{z}_{i}^{m}, \mathbf{z}_{j}^{\pi^{\star}(m)}; 1-\alpha_m)$, where $\text{SLERP}(a, b; t)$ interpolates from a at $t = 0$ to b at $t = 1$, so a larger α_m weights the subnetwork from expert i more heavily. The interpolated latents are assembled into $\widetilde{Z}_{ij}$ and decoded as $\widetilde{W}_{ij} = \mathcal{D}_{\theta}(\widetilde{Z}_{ij})$. The resulting expert then replaces the pair in the compressed MoE layer.

4 Experiments

4.1 Experimental Setup

Models and benchmarks. We conduct experiments on three MoE models ranging 7B-21B parameters: LFM2-8B-A1B (AI, 2025), OLMoE-1B-7B-0125-Instruct (Muennighoff et al., 2025),

Table 1: Downstream performance comparison with existing expert merging methods. All merging methods are evaluated under the same expert reduction rate. The best and second-best results within each expert reduction setting are highlighted in **bold** and underlined, respectively.

		General Knowledge (MCQA)				Mathematics & Coding (Generation)				
Expert Reduction	**Method**	**MMLU**	**ARC-E**	**ARC-C**	**Average ↑**	**GSM8K**	**MATH500**	**MBPP**	**HumanEval**	**Average ↑**
					LFM2-8B-A1B					
0.0%	Vanilla	63.11	75.59	47.63	62.11	81.73	72.40	45.60	70.73	67.62
	VAE-reconstructed	$62.93_{\pm 0.21}$	$75.61_{\pm 0.45}$	$47.41_{\pm 0.61}$	$61.98_{\pm 0.32}$	$82.35_{\pm 0.30}$	$72.20_{\pm 0.64}$	$45.52_{\pm 1.14}$	$71.83_{\pm 3.17}$	$67.97_{\pm 0.72}$
12.5%	F-Merging	55.64	68.18	42.32	55.38	69.14	30.60	03.00	03.66	26.60
	M-SMoE	54.72	72.97	44.11	57.27	57.24	20.40	13.20	25.61	29.11
	HC-SMoE	57.45	69.82	43.87	57.05	64.97	56.00	37.00	57.93	53.98
	Sub-MoE	56.28	69.60	42.27	56.05	70.13	59.60	34.80	54.27	54.70
	SLEM (r=14)	**58.37**	**74.79**	**47.87**	**60.34**	**77.10**	**67.60**	**44.20**	**62.80**	**62.93**
25.0%	F-Merging	47.72	65.03	38.23	50.32	37.83	06.40	00.20	00.61	11.26
	M-SMoE	46.07	66.89	40.31	51.09	02.05	02.20	00.00	00.00	01.06
	HC-SMoE	50.98	60.47	37.15	49.53	54.13	45.20	30.40	44.51	43.56
	Sub-MoE	49.91	64.82	37.75	50.83	43.59	34.20	08.20	28.05	28.51
	SLEM (r=14)	51.10	**70.62**	**42.75**	**54.82**	**70.36**	58.80	**36.20**	**61.59**	**56.74**
	⌞ within-layer merging	**51.29**	67.89	41.47	53.55	68.16	**59.00**	31.40	59.76	54.58
					OLMoE-1B-7B-0125-Instruct					
0.0%	Vanilla	53.45	76.48	47.17	59.03	68.99	23.00	29.60	35.37	39.24
	VAE-reconstructed	$53.21_{\pm 0.10}$	$76.36_{\pm 0.56}$	$47.39_{\pm 0.75}$	$58.98_{\pm 0.32}$	$68.89_{\pm 0.89}$	$23.64_{\pm 1.08}$	$28.40_{\pm 0.89}$	$33.54_{\pm 1.02}$	$38.62_{\pm 0.42}$
12.5%	F-Merging	45.13	66.12	38.91	50.05	41.32	03.60	00.00	00.00	11.23
	M-SMoE	43.56	63.43	35.75	47.58	07.51	03.60	00.00	00.00	02.78
	HC-SMoE	**50.58**	75.49	**48.02**	**58.03**	56.03	09.40	12.00	11.59	22.26
	Sub-MoE	46.26	69.82	39.64	51.91	48.45	07.20	10.60	11.59	19.46
	SLEM (r=16)	47.37	**75.80**	45.90	56.36	**59.74**	**14.20**	**20.00**	**25.00**	**29.74**
25.0%	F-Merging	32.69	62.71	35.32	43.58	05.23	03.60	00.00	00.00	02.21
	M-SMoE	23.16	42.10	26.33	30.53	00.00	00.00	00.00	00.00	00.00
	HC-SMoE	**43.52**	69.41	**41.67**	**51.54**	06.44	03.80	02.40	00.61	03.31
	Sub-MoE	33.56	39.93	22.87	32.12	00.91	01.60	02.00	00.00	01.13
	SLEM (r=16)	30.33	**69.95**	40.87	47.05	**23.12**	**06.60**	**10.40**	**05.49**	**11.40**
					ERNIE-4.5-21B-A3B-PT					
0.0%	Vanilla	73.89	83.30	56.62	71.27	90.67	74.80	73.20	89.63	82.08
	VAE-reconstructed	$74.01_{\pm 0.15}$	$82.74_{\pm 0.33}$	$55.68_{\pm 0.46}$	$70.81_{\pm 0.17}$	$90.42_{\pm 0.59}$	$75.48_{\pm 1.07}$	$72.52_{\pm 0.41}$	$88.78_{\pm 1.13}$	$81.80_{\pm 0.35}$
12.5%	F-Merging	62.46	**82.15**	52.82	65.81	84.69	48.60	36.40	42.07	52.94
	M-SMoE	67.66	74.73	48.36	63.58	82.64	51.60	46.00	51.83	58.02
	HC-SMoE	**70.74**	82.04	**54.61**	**69.13**	**85.90**	59.00	55.40	62.80	65.78
	Sub-MoE	64.83	74.46	44.16	61.15	84.46	52.80	46.80	60.37	61.11
	SLEM (r=12)	65.57	79.92	50.00	65.16	85.75	**65.40**	**64.40**	**83.54**	**74.77**
25.0%	F-Merging	56.92	67.76	40.19	54.96	64.82	21.00	11.80	06.71	26.08
	M-SMoE	54.57	61.45	35.15	50.39	49.20	13.20	10.20	07.32	19.98
	HC-SMoE	**64.53**	**79.06**	**52.35**	**65.32**	**79.83**	37.40	32.20	32.93	45.59
	Sub-MoE	51.42	62.86	35.34	49.87	50.42	18.20	20.20	18.90	26.93
	SLEM (r=12)	53.09	76.26	46.75	58.70	72.25	**44.40**	**46.80**	**51.83**	**53.82**
	⌞ within-layer merging	50.33	75.42	44.80	56.85	59.06	40.00	44.40	45.12	47.15

and ERINE-4.5-21B-A3B-PT (Team, 2025). We evaluate three categories: general knowledge reasoning with MMLU (Hendrycks et al., 2021a), ARC-Easy, and ARC-Challenge (Clark et al., 2018); mathematical reasoning with GSM8K (Cobbe et al., 2021) and MATH500 (Hendrycks et al., 2021b); and code generation with MBPP (Austin et al., 2021) and HumanEval (Chen et al., 2021). General knowledge reasoning is evaluated in an MCQA format, while the other two categories use generation-based evaluation.

Baselines. We compare SLEM against F-Merging, M-SMoE (Li et al., 2024), HC-SMoE (Chen et al., 2025), and Sub-MoE (Li et al., 2026). F-Merging uses expert activation frequencies collected from calibration data to guide the merging process. For all calibration-dependent baselines, we use randomly sampled 32×2048 tokens from C4 dataset (Dodge et al., 2021) to obtain the required routing statistics or expert outputs.

Due to computational constraints, we conduct experiments using greedy decoding with a single run unless otherwise specified. Further details on Expert VAE training, baseline descriptions, and benchmark configurations are provided in Appendix B and C.

4.2 Functional Fidelity of Reconstructed Experts

We first evaluate whether the trained hyperspherical VAE can effectively represent and reconstruct expert weights before applying latent-space merging. We sample latent representations from the posterior conditioned on the pretrained expert weights and decode them back into the corresponding

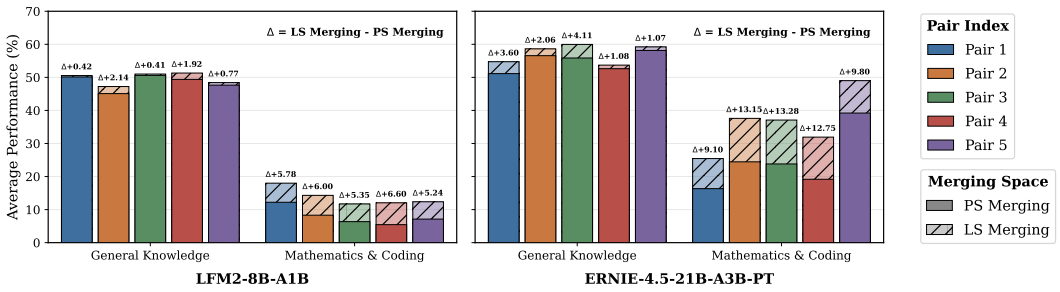


Figure 4: Comparison of parameter-space (PS) and latent-space (LS) merging under identical expert pairs and merging coefficients. Across both models and task categories, LS Merging consistently outperforms PS Merging for every evaluated pair.

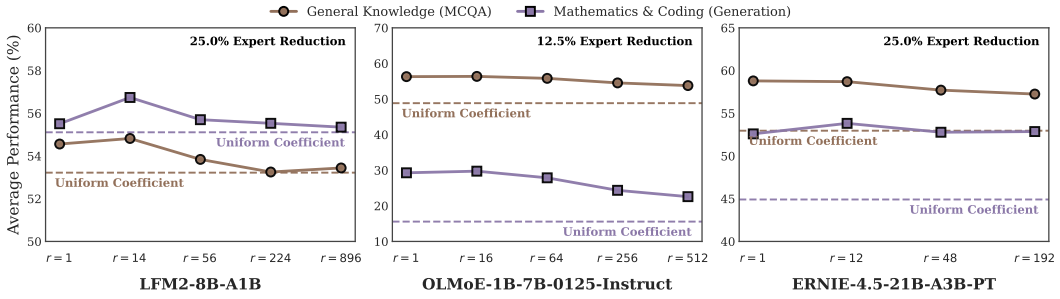


Figure 5: Effect of the truncation rank r for spectrum-aware merging coefficients. Spectrum-aware coefficients consistently outperform the uniform-coefficient baseline across models and tasks.

weights. Table 1 reports the mean and standard deviation of downstream performance over five posterior samples, where the original expert weights are replaced with VAE-reconstructed weights. Across all evaluated tasks, the reconstructed models achieve performance close to that of the original models with modest variations across samples. These results indicate that the learned latent representations retain the information necessary to recover functionally effective expert weights, supporting the use of the learned hyperspherical latent space for subsequent expert interpolation and merging.

4.3 Comparison with Expert Merging Baselines

Table 1 compares SLEM with existing expert merging methods across three MoE models and two expert reduction rates. For LFM2, SLEM achieves the highest average performance on both general knowledge and generation benchmarks at 12.5% and 25% expert reduction. For OLMoE and ERNIE-4.5, HC-SMoE performs better on average for general knowledge benchmarks, while SLEM remains competitive. In contrast, SLEM consistently achieves the highest average generation performance across all models and expert reduction settings.

Overall, these results show that SLEM maintains comparable or better general knowledge performance while providing consistently stronger performance on generation tasks, demonstrating its effectiveness across different model architectures, scales, and compression levels.

4.4 Ablation Study

We ablate the three key components of SLEM: merging in the learned latent space, spectrum-aware merging coefficients, and cross-layer merging.

Merging space. To isolate the effect of the merging space, we randomly construct expert pairs and assign the same uniform coefficients of $[0.5, 0.5]$, varying only whether interpolation is performed in parameter space or in the learned latent space. We repeat this experiment with five independently sampled expert pairs. As shown in Figure 4, latent-space merging consistently outperforms

Table 2: Downstream performance under constrained maximum expert group size. Limiting the group size generally degrades performance compared with unconstrained grouping.

Max Group Size	Method	General Knowledge (MCQA)				Mathematics & Coding (Generation)				
		MMLU	ARC-E	ARC-C	Average ↑	GSM8K	MATH500	MBPP	HumanEval	Average ↑
LFM2-8B-A1B										
∞	M-SMoE	46.07	66.89	40.31	51.09	02.05	02.20	00.00	00.00	01.06
	HC-SMoE	50.98	60.47	37.15	49.53	54.13	45.20	30.40	44.51	43.56
	Sub-MoE	49.91	64.82	37.75	50.83	43.59	34.20	08.20	28.05	28.51
2	M-SMoE	34.55	65.99	38.99	46.51↓ 4.58	03.20	08.34	00.00	00.00=	02.89↑ 1.83
	HC-SMoE	49.84	66.46	40.10	52.13↑ 2.60	01.52	06.00	01.20	01.22	02.49↓ 41.07
	Sub-MoE	47.33	63.93	37.12	49.46↓ 1.37	44.20	31.40	08.00	03.05	21.66↓ 6.85
	SLEM (r=14)	**51.10**	**70.62**	**42.75**	**54.82**	**70.36**	**58.80**	**36.20**	**61.59**	**56.74**
ERNIE-4.5-21B-A3B-PT										
∞	M-SMoE	54.57	61.45	35.15	50.39	49.20	13.20	10.20	07.32	19.98
	HC-SMoE	**64.53**	**79.06**	**52.35**	**65.32**	**79.83**	37.40	32.20	32.93	45.59
	Sub-MoE	51.42	62.86	35.34	49.87	50.42	18.20	20.20	18.90	26.93
2	M-SMoE	54.43	60.48	34.13	49.68↓ 0.71	43.37	11.60	08.80	08.54	18.08↓ 1.90
	HC-SMoE	**55.80**	69.53	41.30	55.54↓ 9.78	45.26	12.60	09.20	11.59	19.66↓ 25.95
	Sub-MoE	47.88	62.29	31.82	47.33↓ 2.54	48.82	17.80	20.80	15.24	25.67↓ 1.26
	SLEM (r=12)	53.09	**76.26**	**46.75**	**58.70**	**72.25**	**44.40**	**46.80**	**51.83**	**53.82**

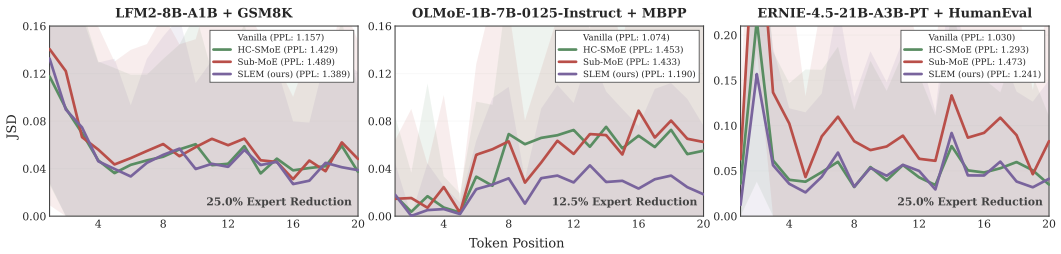


Figure 6: Token-level Jensen–Shannon divergence (JSD) and perplexity (PPL) of merged models on sequences generated by the original model. We evaluate 100 sequences only over the response tokens. Lower values indicate better preservation of the original model's generative behavior.

parameter-space merging for every pair across both benchmark categories and the improvement is particularly pronounced on generation-based benchmarks. These results provide empirical evidence that interpolating experts in the learned latent space offers a more effective merging space than direct interpolation of their parameters.

Spectrum-aware coefficients. We next evaluate spectrum-aware merging coefficients by varying the truncation rank r used for spectral importance estimation and comparing against uniform coefficients of $[0.5, 0.5]$. As shown in Figure 5, spectrum-aware merging consistently outperforms uniform averaging across models and task categories, supporting the effectiveness of spectral importance for determining subnetwork-specific contributions. Performance also varies with truncation rank r, revealing a trade-off in how concentrated the resulting importance distribution becomes. At very small ranks, the normalized SVD importance tends to become relatively uniform across subnetworks, making reducing the benefit of spectrum-aware weighting. As r increases, the importance distribution becomes more discriminative and performance generally improves. However, an overly concentrated distribution can assign negligible importance to some subnetworks, which may discard useful contributions and degrade performance. This suggests that effective merging requires a balance between distinguishing important subnetworks and preserving the contribution of overall subnetworks. We provide the normalized spectral importance distributions under different truncation ranks in Appendix H.

Cross-layer merging. To verify the effectiveness of cross-layer merging, we compare SLEM with a within-layer variant in which the expert-grouping scope is restricted to experts from the same MoE layer, while keeping the remaining components unchanged. As shown in Table 1, allowing cross-layer expert pairs generally improves downstream performance across models and tasks. This suggests that the shared latent space from the trained Expert VAE provides a more flexible representation for identifying compatible experts across layer boundaries.

4.5 Extended Analysis

We further analyze SLEM from two perspectives. First, we examine how constraining the maximum expert group size affects existing merging methods, providing a controlled comparison under the pairwise merging setting used by SLEM. Second, we evaluate how closely the merged models preserve the generative behavior of the original model using token-level Jensen–Shannon divergence (JSD) and perplexity (PPL).

4.5.1 Effect of Expert Group Size Constraints

Consider a pool of N experts partitioned into M merge groups $\{\mathcal{G}_q\}_{q=1}^{M}$, where each group is replaced by a single merged expert. For a fixed expert reduction R,

$$R = \sum_{q=1}^{M} \big(|\mathcal{G}_q| - 1\big), \qquad N_{\text{preserved}} = N - \sum_{q=1}^{M} |\mathcal{G}_q| = N - R - M. \tag{6}$$

Allowing larger groups therefore achieves the same reduction with fewer merge groups, leaving more original experts untouched. Pairwise merging, in contrast, requires $M = R$, which gives $N_{\text{preserved}} = N - 2R$ and involves substantially more original experts in the merging process. This raises the question of whether effective expert merging benefits more from preserving many original experts or from limiting each merged group to few constituents. To answer it, we restrict the maximum group size of existing merging methods to two while keeping the expert reduction rate fixed.

As shown in Table 2, this constraint generally degrades performance across the baselines, with particularly large drops for HC-SMoE. Lasby et al. (2026) reports a high proportion of singleton clusters in HC-SMoE. We hypothesize that limiting the maximum group size reduces the number of preserved experts, which may contribute to the observed performance drop. In contrast, SLEM is designed explicitly around pairwise expert merging and maintains strong performance under the same constraint. These results suggest that SLEM can preserve useful expert functionality even when a larger fraction of the original experts must participate in merging.

4.5.2 Preservation of Generative Behavior

We further examine whether the strong downstream performance of SLEM corresponds to better preservation of the original model's generative behavior. Figure 6 reports token-level JSD between the output distributions of the original and merged models, together with PPL on sequences generated by the original model. Across the evaluated model–task pairs, SLEM exhibits consistently low JSD and achieves the lowest PPL among the merged models, indicating closer agreement with the original model's token distributions. These results suggest that SLEM more faithfully preserves the generative behavior of the original model.

5 Conclusion

In this work, we proposed **Spectrum-Aware Latent Expert Merging (SLEM)**, a calibration-free framework for MoE compression that performs expert merging in a learned latent space. SLEM represents experts across different MoE layers in a shared latent space, enabling both within- and cross-layer merging through geometry-aware interpolation. Expert pairs are selected based on latent similarity, while subnetwork-specific coefficients are derived directly from the spectrum of pretrained expert weights.

Across MoE models of different architectures and parameter scales, SLEM consistently strong performance on mathematical reasoning and code generation tasks. Our ablations further demonstrate the effectiveness of latent-space merging, spectrum-aware coefficients, and cross-layer merging In addition, SLEM more closely preserves the generative behavior of the original model. Overall, these results highlight the potential of learned expert representations and the spectral structure of expert weights for calibration-free MoE compression.

AI USE STATEMENT

Generative AI tools were used solely for language editing, improving the clarity and readability of the manuscript, and assisting with literature search. All research ideas, methodological design, experiments, analyses, and conclusions were developed and verified by the authors.

REPRODUCIBILITY STATEMENT

To support reproducibility, we detail the relevant hyperparameters throughout the Appendix. Our code and merged models will be open-sourced upon acceptance.

ETHICS STATEMENT

All models and datasets used in this work are open-source and publicly available. We thank those who produced, compiled, cleaned, and released them. We have cited their work appropriately and complied with all relevant licensing and usage terms. To the best of our knowledge, these models and datasets do not reveal personally identifiable information or violate copyright restrictions.

REFERENCES

Liquid AI. Lfm2 technical report. *arXiv preprint arXiv:2511.23404*, 2025.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021.

Sikai Bai, Haoxi Li, Jie ZHANG, Zicong Hong, and Song Guo. Diep: Adaptive mixture-of-experts compression through differentiable expert pruning. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 56090–56115. Curran Associates, Inc., 2025. doi: 10.52202/085713-1878.

Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. A Survey on Mixture of Experts in Large Language Models . *IEEE Transactions on Knowledge & Data Engineering*, 37(07):3896–3915, July 2025. ISSN 1558-2191.

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 *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 8594–8620. PMLR, 13–19 Jul 2025.

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

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018.

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

Tim R. Davidson, Luca Falorsi, Nicola De Cao, Thomas Kipf, and Jakub M. Tomczak. Hyperspherical variational auto-encoders. In *34th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2018.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 1286–1305, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.98.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024.

Hao Gu, Wei Li, Lujun Li, Zhu Qiyuan, 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, 13–19 Jul 2025.

Saumya Gupta, Scott Biggs, Moritz Laber, Zohair Shafi, Robin Walters, and Ayan Paul. Deepweightflow: Re-basined flow matching for generating neural network weights. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=fOwsr1VTi8.

Xiaolong Han, Zehong Wang, Bo Zhao, Binchi Zhang, Jundong Li, Damian Borth, Rose Yu, Haggai Maron, Yanfang Ye, Lu Yin, and Ferrante Neri. A survey of weight space learning: Understanding, representation, and generation, 2026. URL https://arxiv.org/abs/2603.10090.

Joëlle Hanna, Damian Falk, Stella X. Yu, and Damian Borth. Geosane: Learning geospatial representations from models, not data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 27804–27814, June 2026.

Shwai He, Run-Ze Fan, Liang Ding, Li Shen, Tianyi Zhou, and Dacheng Tao. Merging experts into one: Improving computational efficiency of mixture of experts. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://openreview.net/forum?id=4KRiWsfOwn.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021a.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *Advances in neural information processing systems*, 2021b.

Vincent Herrmann, Francesco Faccio, and Jürgen Schmidhuber. Learning useful representations of recurrent neural network weight matrices, 2024. URL https://arxiv.org/abs/2403.11998.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Sy2fzU9gl.

Wentao Hu, Mingkuan Zhao, Shuangyong Song, Xiao yi Zhu, Xin Lai, and Jiayin Wang. Mosaic pruning: A hierarchical framework for generalizable pruning of mixture-of-experts models. In *AAAI Conference on Artificial Intelligence*, 2026.

Weizhong Huang, Yuxin Zhang, Xiawu Zheng, Fei Chao, Rongrong Ji, and Liujuan Cao. Discovering important experts for mixture-of-experts models pruning through a theoretical perspective. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=7kQjbCQwtT.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=6t0Kwf8-jrj.

Jiwoo Kim, Swarajh Mehta, Hao-Lun Hsu, Hyunwoo Ryu, Yudong Liu, and Miroslav Pajic. Nnit: Width-agnostic neural network generation with structurally aligned weight spaces, 2026. URL https://arxiv.org/abs/2603.00180.

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, Xiaoyu Qin, Wei Li, Hao Gu, Sirui Han, and Yike Guo. Sub-moe: Efficient mixture-of-expert llms compression via subspace expert merging. In *AAAI 2026*, 2026.

Pingzhi Li, Zhenyu Zhang, Prateek Yadav, Yi-Lin Sung, Yu Cheng, Mohit Bansal, and Tianlong Chen. Merge, then compress: Demystify efficient smoe with hints from its routing policy, 2024. URL https://arxiv.org/abs/2310.01334.

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, 13–19 Jul 2025.

Jiacheng Liu, Peng Tang, Wenfeng Wang, Yuhang Ren, Xiaofeng Hou, Pheng Ann Heng, Minyi Guo, and Chao Li. A survey on inference optimization techniques for mixture of experts models. *ACM Computing Surveys*, 58(10):1–37, March 2026. ISSN 1557-7341. doi: 10.1145/3794845.

Ruijie Miao, Yilun Yao, Zihan Wang, Zhiming Wang, Bairen Yi, LingJun Liu, Yikai Zhao, and Tong Yang. Mergemoe: Efficient compression of moe models via expert output merging, 2025. URL https://arxiv.org/abs/2510.14436.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Evan Pete Walsh, Oyvind Tafjord, Nathan Lambert, Yuling Gu, Shane Arora, Akshita Bhagia, Dustin Schwenk, David Wadden, Alexander Wettig, Binyuan Hui, Tim Dettmers, Douwe Kiela, Ali Farhadi, Noah A. Smith, Pang Wei Koh, Amanpreet Singh, and Hannaneh Hajishirzi. OLMoe: Open mixture-of-experts language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

Dung Viet Nguyen, Anh Thi, Minh Hoang Nguyen, Luc Nguyen, Shiqi Jiang, Ethan Fetaya, Duy Linh Tran, Gal Chechik, and Tan Nguyen. Expert merging in sparse mixture of experts with nash bargaining. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 148047–148075, 2026.

Viet Dung Nguyen, Minh Nguyen Hoang, Luc Nguyen, Rachel Teo, Tan Minh Nguyen, and Linh Duy Tran. CAMEx: Curvature-aware merging of experts. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=nT2u0M0nf8.

Konstantin Schürholt, Dimche Kostadinov, and Damian Borth. Self-supervised representation learning on neural network weights for model characteristic prediction. In *Advances in Neural Information Processing Systems*, volume 34, pp. 16481–16493. Curran Associates, Inc., 2021.

Konstantin Schürholt, Boris Knyazev, Xavier Giró-i Nieto, and Damian Borth. Hyper-representations as generative models: sampling unseen neural network weights. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Noam Shazeer, *Azalia Mirhoseini, *Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017.

Bedionita Soro, Bruno Andreis, Hayeon Lee, Wonyong Jeong, Song Chong, Frank Hutter, and Sung Ju Hwang. Diffusion-based neural network weights generation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=j8WHjM9aMm.

Bedionita Soro, Aoxuan Silvia Zhang, Bruno Andreis, Jaehyeong Jo, Song Chong, and Sung Ju Hwang. LS-merge: Merging language models in latent space. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=VSDV0SWwOC.

Baidu ERNIE Team. Ernie 4.5 technical report, 2025.

Viet-Hoang Tran, Thieu Vo, An Nguyen, Tho-Huu Tran, Minh-Khoi Nguyen-Nhat, Thanh Tran, Duy-Tung Pham, and Tan Nguyen. Equivariant neural functional networks for transformers. In *International Conference on Learning Representations*, volume 2025, pp. 16270–16341, 2025.

Zhangyang Wang, Peihao Wang, and Kai Wang. Position: Weight space should be a first-class generative AI modality. In *Forty-third International Conference on Machine Learning Position Paper Track*, 2026. URL https://openreview.net/forum?id=vu0ousWLde.

Yanyue Xie, Zhi Zhang, Ding Zhou, Cong Xie, Ziang Song, Xin Liu, Yanzhi Wang, Xue Lin, and An Xu. Moe-pruner: Pruning mixture-of-experts large language model using the hints from its router, 2024. URL https://arxiv.org/abs/2410.12013.

Jiacheng Xu and Greg Durrett. Spherical latent spaces for stable variational autoencoders. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 4503–4513, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1480. URL https://aclanthology.org/D18-1480/.

Yuzhuang Xu, Xu Han, Yuanchi Zhang, Yixuan Wang, Yijun Liu, Shiyu Ji, Qingfu Zhu, and Wanxiang Che. Camera: Multi-matrix joint compression for moe models via micro-expert redundancy analysis, 2025. URL https://arxiv.org/abs/2508.02322.

Geng Zhang, Han Yuxuan, Yuxuan Lou, Yiqi Zhang, Wangbo Zhao, and Yang You. Mone: Replacing redundant experts with lightweight novices for structured pruning of moe. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 39983–40009, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/431c7c0b59aeafa2182b4d6617884b53-Paper-Conference.pdf.

Yushu Zhao, Zheng Wang, and Minjia Zhang. Puzzlemoe: Efficient compression of large mixture-of-experts models via sparse expert merging and bit-packed inference. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=D7pFsC7UAr.

Allan Zhou, Kaien Yang, Kaylee Burns, Adriano Cardace, Yiding Jiang, Samuel Sokota, J. Zico Kolter, and Chelsea Finn. Permutation equivariant neural functionals, 2023a. URL https://arxiv.org/abs/2302.14040.

Allan Zhou, Kaien Yang, Yiding Jiang, Kaylee Burns, Winnie Xu, Samuel Sokota, J. Zico Kolter, and Chelsea Finn. Neural functional transformers. In *Advances in Neural Information Processing Systems*, volume 36, pp. 77485–77502. Curran Associates, Inc., 2023b. doi: 10.52202/075280-3387.

Zeliang Zong, Kai Zhang, Yarong Wang, Wenming Tan, Ye Ren, and Jilin Hu. Less token, more signal: Moe expert pruning via critical token selection. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=4iupzej9nT.

A Limitations and Future Directions

Compute constraints limit our evaluation to models with up to 21B parameters and fixed-budget experiments. In addition, as with other MoE compression methods, although SLEM recovers part of the performance lost through compression, the resulting compressed checkpoints do not yet fully match the performance of their uncompressed pretrained counterparts at higher expert reduction rates.

SLEM also introduces additional design and computational costs. It requires training an expert VAE before merging, which incurs additional computation and resource overhead. Moreover, the rank used for spectral importance estimation is a hyperparameter whose optimal value can vary across models and tasks. Reducing the need for model-specific rank selection and improving the efficiency of latent-space training are therefore important directions for future work.

Finally, the current expert grouping strategy is restricted to one-to-one expert pairs. This pairwise formulation limits the achievable compression under a single merging stage and does not directly support consolidating three or more experts into a single group. Extending SLEM to many-to-one merging will require grouping criteria and aggregation strategies that can characterize the compatibility and relative contributions of multiple experts jointly. Developing such group-level similarity measures and merging rules could enable higher compression ratios while preserving expert functionality.

B Experimental Details

We provide further details on the models, baseline implementations, and evaluation protocols used in our experiments.

B.1 Models

LFM2-8B-A1B (AI, 2025). LFM2-8B-A1B contains 8.3B total parameters, with approximately 1.5B parameters activated per token. Among its 24 layers, the first two employ dense feed-forward layers and the remaining 22 use sparse MoE layers. Each MoE layer consists of 32 routed experts and activates the top-4 experts for each token. The model hidden dimension is $D = 2048$, and each routed expert has an intermediate dimension of $D_{\mathrm{MoE}} = 1792$.

OLMoE-1B-7B-0125-Instruct (Muennighoff et al., 2025). OLMoE-1B-7B-0125-Instruct has 6.9B total parameters and approximately 1.3B active parameters per token. All 16 layers employ sparse MoE feed-forward networks without shared experts, each containing 64 routed experts with top-8 routing. Its hidden dimension is $D = 2048$, with an expert intermediate dimension of $D_{\mathrm{MoE}} = 1024$.

ERNIE-4.5-21B-A3B-PT (Team, 2025). ERNIE-4.5-21B-A3B-PT is a 21B-parameter MoE model with approximately 3B active parameters per token. It consists of 28 layers, with the first layer using a dense feed-forward network and the remaining 27 employing MoE layers. Each MoE layer contains 64 routed experts together with two shared experts, and selects the top-6 routed experts for each token. The model uses a hidden dimension of $D = 2560$ and an MoE intermediate dimension of $D_{\mathrm{MoE}} = 1536$.

B.2 Baselines

M-SMoE (Li et al., 2024). M-SMoE performs routing-guided expert grouping followed by frequency-weighted parameter merging. It selects dominant experts according to their activation frequencies from the calibration set and assigns the remaining experts to the most similar dominant expert. Expert similarity is estimated from router logits collected over the calibration set:

$$\mathcal{S}_{\mathrm{router}}(i, j) = \cos(\mathbf{r}_i, \mathbf{r}_j)\,, \tag{7}$$

where $\mathbf{r}_i$ denotes the router-logit vector of expert i over the calibration tokens. Given a resulting group $\mathcal{C}$, the hidden units of different experts are aligned through neuron permutation matching, and

then experts are merged as

$$\widetilde{W}_{\mathcal{C},p} = \frac{\sum_{i\in\mathcal{C}} f_i W_{i,p}}{\sum_{i\in\mathcal{C}} f_i}, \tag{8}$$

where f_i is the activation frequency of expert i and $p \in \mathcal{P}$ denotes the projection type.

HC-SMoE (Chen et al., 2025). HC-SMoE groups experts according to their similarity measured from expert outputs. Given a calibration set $\mathcal{D}_{\text{cal}}$ containing T tokens, each expert is represented by its average output:

$$\mathbf{o}_i = \frac{1}{T} \sum_{x\in\mathcal{D}_{\text{cal}}} E_i(x). \tag{9}$$

Starting from singleton expert clusters, it performs agglomerative hierarchical clustering using the distance between these representations. The distance between two clusters $\mathcal{C}_a$ and $\mathcal{C}_b$ is

$$d(\mathcal{C}_a, \mathcal{C}_b) = \frac{1}{|\mathcal{C}_a||\mathcal{C}_b|} \sum_{i\in\mathcal{C}_a} \sum_{j\in\mathcal{C}_b} \|\mathbf{o}_i - \mathbf{o}_j\|_2. \tag{10}$$

The closest clusters are iteratively merged until the target number of experts is reached, and the experts within each cluster are merged according to equation 8.

Sub-MoE (Li et al., 2026). Sub-MoE groups functionally similar experts using their outputs on the calibration set. For a pair of experts, the functional similarity is defined as the average cosine similarity:

$$\mathcal{S}_{\text{out}}(i,j) = \frac{1}{T} \sum_{x\in\mathcal{D}_{\text{cal}}} \frac{E_i(x)^\top E_j(x)}{\|E_i(x)\|_2 \|E_j(x)\|_2}, \tag{11}$$

which is used to form expert clusters through K-means clustering. After, the experts within each cluster are aligned in a shared subspace. For projection p, the weight matrices of experts in a cluster $\mathcal{C}$ are jointly decomposed as

$$\text{SVD}\left([W_{i_1,p}; \ldots; W_{i_{|\mathcal{C}|},p}]\right) = U_p \Sigma_p [V_{i_1,p}; \ldots; V_{i_{|\mathcal{C}|},p}]^\top. \tag{12}$$

The expert-specific V components are then combined according to their activation frequencies,

$$\widetilde{V}_{\mathcal{C},p} = \frac{\sum_{i\in\mathcal{C}} f_i V_{i,p}}{\sum_{i\in\mathcal{C}} f_i}, \tag{13}$$

and the merged projection is reconstructed in the shared subspace as

$$\widetilde{W}_{\mathcal{C},p} = U_p \Sigma_p \widetilde{V}_{\mathcal{C},p}^\top. \tag{14}$$

Frequency-based Merging. Frequency-based Merging (F-Merging) uses expert activation frequency as both the grouping and merging signal. Given a calibration set, we compute the activation frequency of each expert. Experts are then paired in a one-to-one manner by matching the most frequently activated expert with the least frequently activated expert. The selected pair are merged according to equation 8, and remove both experts from further pairing. This procedure is repeated until the target number of experts is reached.

We do not include PuzzleMoE (Zhao et al., 2026) because its deployment pipeline differs substantially from the standard expert setting considered in our experiments. We also exclude NAMEx (Nguyen et al., 2026) because its original formulation does not target reducing the number of stored experts under a fixed expert-reduction budget, making it not directly comparable to our compression setting.

B.3 Evaluation Protocols

We conduct all evaluations using `lm_eval` framework (Gao et al., 2024). For MCQA tasks, each question and its candidate options are provided as input, and the option with the highest probability is selected as the prediction. For generation-based tasks, the model generates responses autoregressively, after which the final answer is extracted using task-specific regular expressions. The benchmark-specific evaluation settings are summarized in Table 3. All experiments are conducted on a single NVIDIA RTX PRO 6000 Blackwell Max-Q GPU (96GB) using bfloat16 precision without quantization. Due to computational constraints, we use greedy decoding and report results from a single evaluation run unless otherwise specified.

Table 3: Benchmark-specific evaluation settings, including the evaluation metric, few-shot setting, maximum sequence length, and maximum generation tokens.

Task	Metric	Few-shot	Maximum Seqence Length	Maximum Generation Tokens
General Knowledge (MCQA)				
MMLU (Hendrycks et al., 2021a)	Accuracy	0-shot	4,096	N/A
ARC-Easy (Clark et al., 2018)	Accuracy	0-shot	4,096	N/A
ARC-Easy (Clark et al., 2018)	Accuracy	0-shot	4,096	N/A
Mathematics (Generation)				
GSM8K (Cobbe et al., 2021)	Exact Match	8-shot (single-turn)	4,096	1,024
MATH (Hendrycks et al., 2021b)	Exact Match	4-shot (single-turn)	4,096	1,024
Coding (Generation)				
MBPP (Austin et al., 2021)	Pass@1	3-shot (single-turn)	4,096	1,024
HumanEval (Chen et al., 2021)	Pass@1	0-shot	4,096	1,024

Table 4: Architecture configurations of the hyperspherical expert VAE for each evaluated model. Training and optimization hyperparameters are provided in Appendix C.

Model	# layers (enc./dec.)	$D_{\{\text{emb, hidden, latent}\}}$	D_{FFN}	# attn. heads	Pre-LN architecture
LFM2-8B-A1B	6	3,072	$4 \times 3{,}072$	24	yes
OLMoE-1B-7B-0125-Instruct	6	3,072	$4 \times 3{,}072$	24	yes
ERNIE-4.5-21B-A3B-PT	6	3,840	$4 \times 3{,}840$	30	yes

C Hyperspherical Expert VAE Training

We provide further details on the hyperspherical expert VAE training, including model-specific architecture configurations, training and optimization hyperparameters, and the computational resources used for training.

Training configurations. Table 4 summarizes the model-specific architecture configurations of the hyperspherical Expert VAE, including the number of encoder and decoder layers, representation dimensions, feed-forward dimensions, and attention heads. All models use a transformer-based encoder and decoder with a Pre-LN architecture. For training, we use a von Mises–Fisher (vMF) latent distribution with concentration parameter $\kappa = 524{,}288$ and KL weight $\beta = 10^{-3}$. Expert weights are standardized independently within each MoE layer, and layer-specific priors are initialized from the same randomly sampled direction. We train each VAE for 200 epochs with a batch size of 4, learning rate 5×10^{-5}, weight decay 10^{-6}, and gradient clipping at 1.0. We use a cosine learning rate scheduler and bfloat16 precision training.

Computational resources. We train each Expert VAE on a single NVIDIA H200 GPU (140GB). The training cost depends on the model size and VAE configuration; for the largest evaluated model, ERNIE-4.5-21B-A3B-PT, training takes ≈ 18.7 hours.

D Analysis of SLEM Design Variants

We analyze alternative design choices of SLEM. Specifically, we compare variants that modify the scope of expert grouping and merging coefficient schemes.

D.1 Expert Grouping Variants

SLEM constrains the maximum number of merged experts assigned to each layer, preventing merge pairs from being overly concentrated in a small subset of layers. Here, we further examine the role of this layer-wise constraint by removing the maximum merged-expert limit. We consider two variants: within-layer adaptive merging, which selects expert pairs only within each layer, and cross-layer adaptive merging, which allows pairs to be selected across layers.

As shown in Table 5, removing the layer-wise constraint generally degrades performance across the three models, with a particularly large drop for within-layer adaptive merging. We hypothesize

Table 5: Comparison of SLEM design variants for expert grouping and merging coefficients. We evaluate alternative grouping strategies, uniform and row-norm coefficients, and projection-specific spectral importance using only gate, up, or down projections. All variants are compared with the default SLEM under the same expert reduction setting.

Expert Reduction	Method	General Knowledge (MCQA)				Mathematics & Coding (Generation)				
		MMLU	ARC-E	ARC-C	Average ↑	GSM8K	MATH500	MBPP	HumanEval	Average ↑
LFM2-8B-A1B										
25.0%	SLEM (r=14)	51.10	70.62	42.75	54.82	70.36	58.80	36.20	61.59	56.74
	∟ within-layer adaptive	53.12	67.85	41.81	54.26↓ 0.56	68.84	60.00	22.00	43.90	48.69↓ 8.05
	∟ cross-layer adaptive	52.98	69.19	43.60	55.26↑ 0.44	68.39	60.40	34.40	55.49	54.67↓ 2.07
	∟ uniform coefficient	50.48	68.01	41.47	53.32↓ 1.50	68.92	58.80	35.40	57.32	55.11↓ 1.63
	∟ row-norm coefficient	50.71	68.39	41.47	53.52↓ 1.30	67.70	60.20	35.40	57.32	55.15↓ 1.59
	∟ spectral importance: gate-only	52.14	69.11	41.04	54.10↓ 0.72	69.83	59.80	30.20	51.83	52.91↓ 3.83
	∟ spectral importance: up-only	51.58	69.74	42.32	54.55↓ 0.27	68.69	59.80	30.40	54.88	53.44↓ 3.30
	∟ spectral importance: down-only	51.47	67.68	41.55	53.57↓ 1.25	71.49	57.80	36.20	62.20	56.92↑ 0.18
OLMoE-1B-7B-0125-Instruct										
12.5%	SLEM (r=16)	47.37	75.80	45.90	56.36	59.74	14.20	20.00	25.00	29.74
	∟ within-layer adaptive	46.25	74.41	44.88	55.18↓ 1.18	49.58	09.80	18.00	19.51	24.22↓ 5.52
	∟ cross-layer adaptive	43.60	75.13	46.08	54.94↓ 1.42	53.83	10.20	19.00	19.51	25.64↓ 4.10
	∟ uniform coefficient	44.25	63.89	38.40	48.84↓ 7.52	38.82	08.20	10.40	04.88	15.57↓ 14.17
	∟ row-norm coefficient	46.02	71.97	42.32	53.44↓ 2.92	50.87	12.00	15.80	14.02	23.17↓ 6.57
	∟ spectral importance: gate-only	47.02	74.49	44.80	55.44↓ 0.92	59.44	11.80	21.00	21.95	28.55↓ 1.19
	∟ spectral importance: up-only	47.77	75.80	45.56	56.38↑ 0.02	58.38	14.00	22.60	21.95	29.23↓ 0.51
	∟ spectral importance: down-only	48.07	76.05	45.22	56.45↑ 0.09	58.15	14.80	18.60	25.61	29.29↓ 0.45
ERNIE-4.5-21B-A3B-PT										
25.0%	SLEM (r=12)	53.09	76.26	46.75	58.70	72.25	44.40	46.80	51.83	53.82
	∟ within-layer adaptive	53.27	76.14	43.26	57.55↓ 1.15	63.08	40.20	42.40	39.63	46.33↓ 7.49
	∟ cross-layer adaptive	51.84	74.92	43.52	56.76↓ 1.94	71.57	42.00	43.40	53.66	52.66↓ 1.16
	∟ uniform coefficient	44.62	73.23	41.04	52.96↓ 5.74	66.11	40.60	40.00	32.93	44.91↓ 8.91
	∟ row-norm coefficient	51.46	74.75	43.86	56.69↓ 2.01	71.27	44.00	47.60	51.83	53.68↓ 0.14
	∟ spectral importance: gate-only	52.86	75.97	44.54	57.79↓ 0.91	71.72	38.20	45.20	50.61	51.43↓ 2.39
	∟ spectral importance: up-only	52.41	74.37	44.03	56.94↓ 1.76	70.43	45.20	46.00	46.95	52.15↓ 1.67
	∟ spectral importance: down-only	54.93	77.40	45.39	59.24↑ 0.54	71.49	42.00	42.20	53.66	52.34↓ 1.48

that this degradation arises from an imbalanced distribution of merge operations across layers. Since expert redundancy, as reflected by latent similarity, is not necessarily uniform across layers, unconstrained similarity-based grouping can concentrate many merges in a small subset of layers, disproportionately reducing their expert capacity while leaving others largely unchanged. This effect is more pronounced for within-layer adaptive merging, where the candidate pool is restricted to each individual layer and highly similar local experts can be repeatedly selected. Cross-layer adaptive merging partially alleviates this issue by providing a broader pool of candidate experts, but still underperforms the balanced grouping used by SLEM. These results suggest that the layer-wise constraint acts as an effective regularizer, preserving balanced expert capacity across the layers while retaining the flexibility of cross-layer merging.

D.2 Merging Coefficient Variants

SLEM determines subnetwork-specific merging coefficients from the spectrum of pretrained expert weights without relying on calibration data. Here, we examine alternative calibration-free coefficient schemes to understand the contribution of spectral truncation and the aggregation of importance across projection types.

Uniform coefficient and row-norm coefficient. A simple calibration-free strategy is to assign equal contributions to the two matched subnetworks. We therefore consider a uniform coefficient of $[0.5, 0.5]$. As another alternative, we use the squared row norm of each projection weight as the subnetwork importance, $s^{\text{norm}}_{i,p,m} = \frac{\|W_{i,p}[m,:]\|_2^2}{\|W_{i,p}\|_F^2}$, followed by the same projection-wise averaging and coefficient computation used in SLEM. By the SVD identity, this corresponds exactly to the full-rank case of our spectral importance.

As shown in Table 5, both uniform and row-norm coefficients generally underperform our spectrum-aware coefficients. The uniform variant ignores differences in subnetwork importance entirely, while the row-norm variant incorporates the full weight energy without distinguishing dominant spectral directions. These results suggest that concentrating importance estimation on the leading singular directions provides a more informative signal for assigning the relative contributions of matched subnetworks.

Table 6: Downstream performance comparison with and without KL regularization during Expert VAE training. The VAE with $\beta = 0$ variant removes the KL divergence term while keeping the remaining training settings unchanged.

Expert Reduction	Method	General Knowledge (MCQA)				Mathematics & Coding (Generation)				
		MMLU	**ARC-E**	**ARC-C**	**Average ↑**	**GSM8K**	**MATH500**	**MBPP**	**HumanEval**	**Average ↑**
LFM2-8B-A1B										
0.0%	Vanilla	63.11	75.59	47.63	62.11	81.73	72.40	45.60	70.73	67.62
	VAE-reconstructed	$62.93_{\pm 0.21}$	$75.61_{\pm 0.45}$	$47.41_{\pm 0.61}$	$61.98_{\pm 0.32}$	$82.35_{\pm 0.30}$	$72.20_{\pm 0.64}$	$45.52_{\pm 1.14}$	$71.83_{\pm 3.17}$	$67.97_{\pm 0.72}$
	⌞ VAE with β=0	$62.94_{\pm 0.19}$	$75.14_{\pm 0.50}$	$47.25_{\pm 0.61}$	$61.78_{\pm 0.19}$	$82.02_{\pm 0.49}$	$72.68_{\pm 1.84}$	$45.56_{\pm 1.51}$	$69.39_{\pm 1.18}$	$67.41_{\pm 0.62}$
12.5%	SLEM (r=14)	58.37	74.79	47.87	60.34	77.10	67.60	44.20	62.80	62.93
	⌞ VAE with β=0	54.32	68.48	42.41	$55.07_{\downarrow 5.27}$	78.54	60.80	29.00	39.63	$51.99_{\downarrow 10.94}$
25.0%	SLEM (r=14)	51.10	70.62	42.75	54.82	70.36	58.80	36.20	61.59	56.74
	⌞ VAE with β=0	44.15	64.27	38.48	$48.97_{\downarrow 5.85}$	68.76	45.40	17.60	32.93	$41.17_{\downarrow 15.57}$
OLMoE-1B-7B-0125-Instruct										
0.0%	Vanilla	53.45	76.48	47.17	59.03	68.99	23.00	29.60	35.37	39.24
	VAE-reconstructed	$53.21_{\pm 0.10}$	$76.36_{\pm 0.56}$	$47.39_{\pm 0.75}$	$58.98_{\pm 0.32}$	$68.89_{\pm 0.89}$	$23.64_{\pm 1.08}$	$28.40_{\pm 0.89}$	$33.54_{\pm 1.02}$	$38.62_{\pm 0.42}$
	⌞ VAE with β=0	$53.37_{\pm 0.16}$	$76.26_{\pm 0.51}$	$47.30_{\pm 1.14}$	$58.98_{\pm 0.53}$	$68.19_{\pm 0.62}$	$22.96_{\pm 0.48}$	$29.28_{\pm 1.13}$	$33.66_{\pm 0.98}$	$38.52_{\pm 0.42}$
12.5%	SLEM (r=16)	47.37	75.80	45.90	56.36	59.74	14.20	20.00	25.00	29.74
	⌞ VAE with β=0	42.68	71.46	44.03	$52.72_{\downarrow 3.74}$	52.08	13.60	17.80	15.24	$24.68_{\downarrow 5.06}$
25.0%	SLEM (r=16)	30.33	69.95	40.87	47.05	23.12	06.60	10.40	05.49	11.40
	⌞ VAE with β=0	33.59	62.12	35.58	$43.76_{\downarrow 3.29}$	14.10	03.60	06.00	01.22	$06.23_{\downarrow 5.17}$

Projection type selection. SLEM computes spectral importance independently for the `gate`, `up`, and `down` projections and averages them to obtain a unified subnetwork importance score. To examine the contribution of this aggregation, we evaluate variants that derive merging coefficients from only one projection type at a time.

Table 5 shows that the effectiveness of each individual projection varies substantially across tasks. These results indicate that each projection captures complementary aspects of subnetwork importance. Averaging their importance scores therefore yields a more balanced estimate, resulting in more robust performance across task categories than relying on any single projection alone.

E Effect of KL Regularization on Expert VAE Training

We analyze how KL regularization during Expert VAE training affects subsequent expert merging. For this purpose, we train an additional VAE with the same posterior sampling procedure but set the KL weight to $\beta = 0$, while keeping the other training configurations unchanged.

As shown in Table 6, both VAEs achieve similar reconstruction performance on LFM2 and OLMoE. However, after latent-expert merging, the VAE trained without KL regularization shows substantially worse downstream performance. This indicates that (i) preserving reconstruction quality alone is not sufficient for effective latent interpolation, and (ii) KL regularization encourages the posterior representations to remain organized around the shared prior, while maintaining reconstruction performance. Consequently, KL regularization yields a latent space where interpolated representations remain more compatible with the decoder, leading to more effective expert merging.

F Paired Bootstrap Confidence Intervals

Since SLEM shows particularly strong performance on generation-based benchmarks involving mathematical reasoning and code generation, we further assess the statistical reliability of these gains on LFM2-8B-A1B. We compare SLEM with HC-SMoE, the strongest competing baseline, using paired bootstrap confidence intervals. For each benchmark, we resample the evaluation instances with replacement while preserving the pairing between the two methods. We perform 200,000 bootstrap resamples and report the 95% confidence interval (CI) of the performance difference in percentage points.

As shown in Table 7, the confidence intervals are consistently positive for GSM8K, MATH500, and MBPP at both expert reduction rates. HumanEval includes zero at 12.5% reduction but becomes clearly positive at 25.0%. The confidence interval of the average generation performance also excludes zero at both reduction rates, supporting the statistical reliability of the overall generation gains of SLEM over HC-SMoE.

Table 7: Paired bootstrap 95% confidence intervals (pp) of the performance difference between SLEM and HC-SMoE on LFM2-8B-A1B generation benchmarks. We use 200,000 bootstrap resamples. Positive intervals favor SLEM.

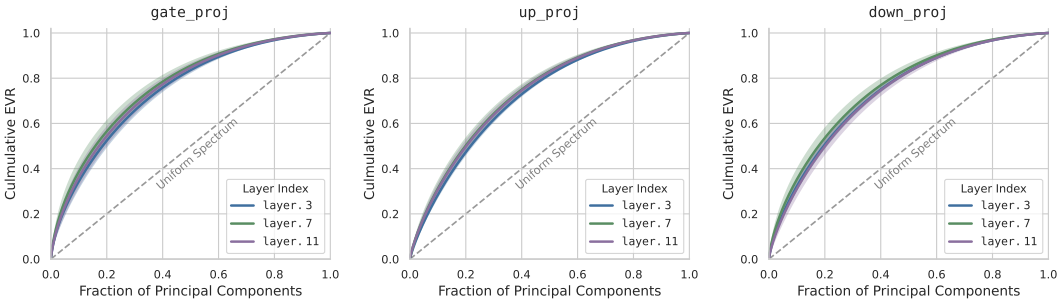
Expert Reduction	**GSM8K** $n = 1{,}319$	**MATH500** $n = 500$	**MBPP** $n = 500$	**HumanEval** $n = 164$	**Average**
12.5%	[9.10, 14.33]	[7.60, 15.60]	[3.40, 11.00]	[-3.05, 12.80]	[6.38, 11.31]
25.0%	[13.19, 18.65]	[9.40, 17.80]	[1.00, 9.80]	[7.93, 23.78]	[10.07, 15.32]

G Cumulative EVR of Expert Weights

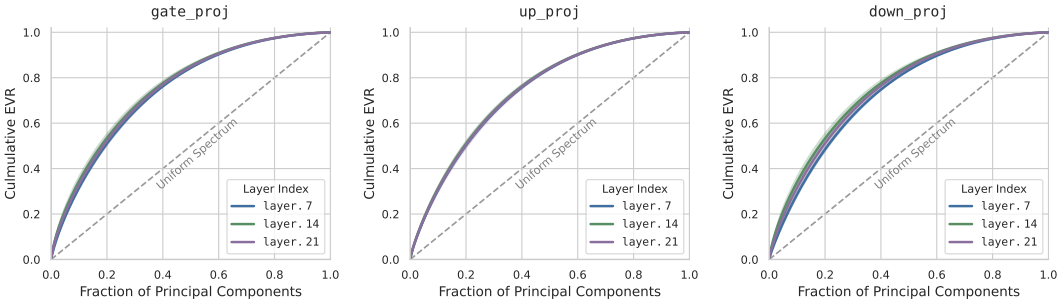
We provide additional cumulative explained variance ratio (EVR) analyses for the other evaluated models. As shown in Figure 7, their expert projection weights exhibit trends similar to those observed in the main text: variance is consistently concentrated along a subset of principal directions rather than being uniformly distributed across the spectrum. This suggests that the dominant spectral structure of expert weights is a common characteristic across the evaluated MoE models, further supporting the use of spectral information for estimating subnetwork importance.

H Distribution of Normalized Spectral Importance

Figure 8 visualizes the normalized spectral importance across subnetworks under different ranks r for each evaluated model. As r varies, the importance distribution changes in concentration, indicating that the truncation rank controls how selectively spectral energy is assigned across subnetworks.



(a) OLMoE-1B-7B-0125-Instruct



(b) ERNIE-4.5-21B-A3B-PT

Figure 7: Cumulative explained variance ratio (EVR) of expert projection weights across layers in other evaluated models.

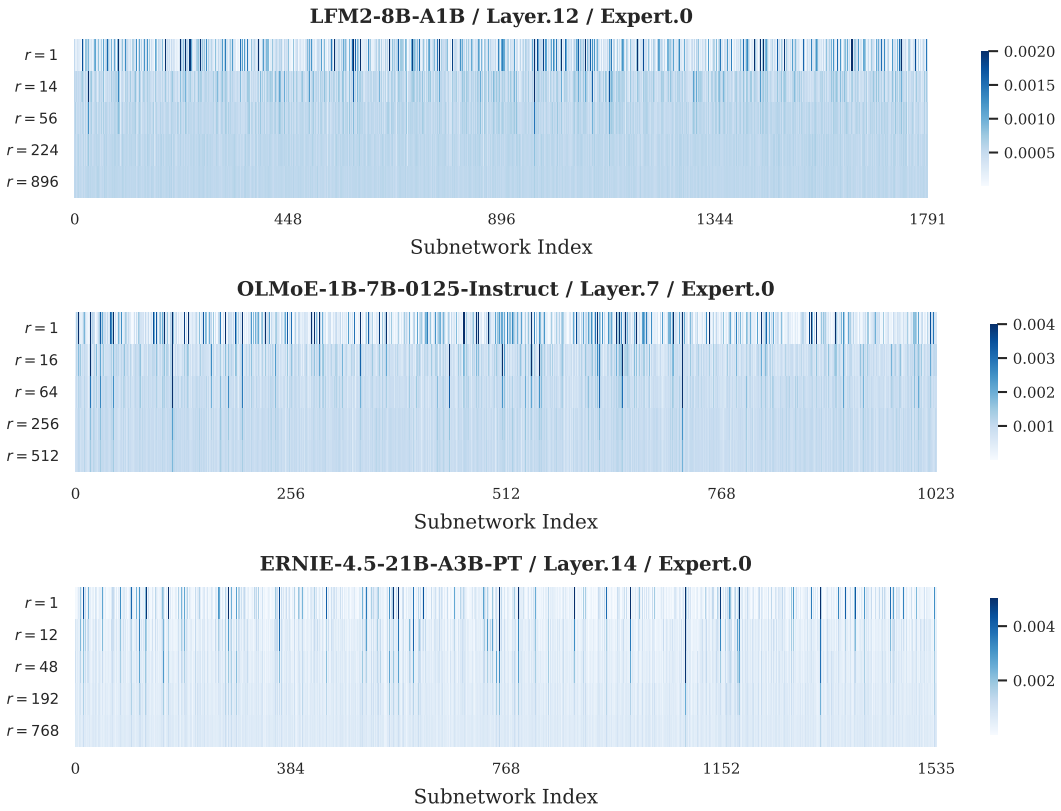


Figure 8: Normalized spectral importance across subnetworks under different truncation ranks r. Lower ranks produce more concentrated importance distributions, while larger ranks distribute importance more uniform across subnetworks.