Eigenvectors of Experts are Training-free Non-collapsing Routers

Giang Do, Hung Le, Truyen Tran

Applied Artificial Intelligence Initiative (A2I2), Deakin University, Victoria, Australia

Content

1. Problems

Routers are a **key component** in Sparse Mixture of Experts (SMoE), but they remain challenging:

- **Difficult to learn:** Router optimization is challenging because routing decisions receive only **indirect** and **sparse gradient signals**.

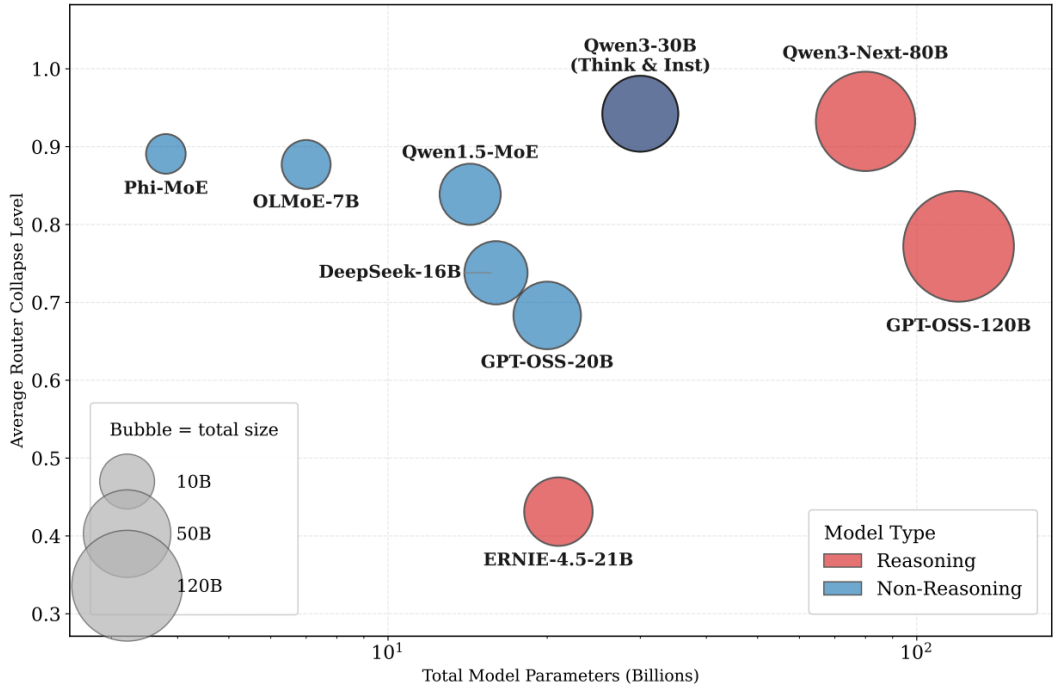


Figure 1. Average router collapse levels across layers for ten state-of-the-art MoE-based LLMs. The results demonstrate that while all models exhibit router collapse, the intensity varies between reasoning and non-reasoning models. Best viewed in color.

1. Problems

Routers are a **key component** in Sparse Mixture of Experts (SMoE), but they remain challenging:

- **Difficult to learn:** Router optimization is challenging because routing decisions receive only **indirect** and **sparse gradient signals**.
- **Router collapse:** Our analysis of 10 **well-trained** large-scale MoEs ranging from 3B to 120B parameters shows severe collapse, with **collapse levels** between **0.4** and **0.9**

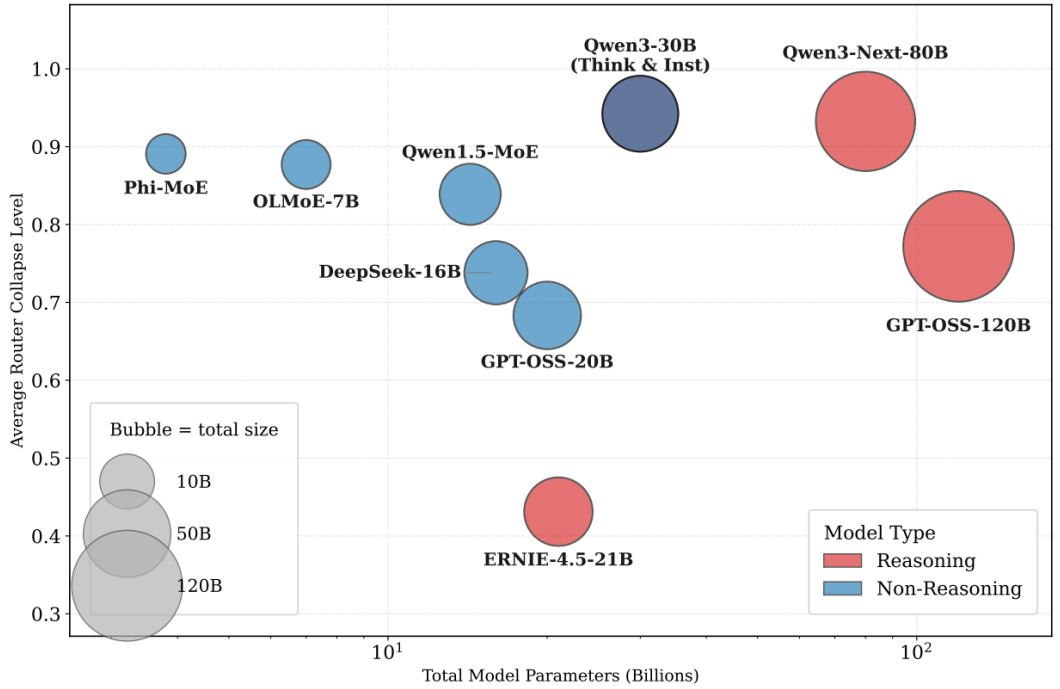


Figure 1. Average router collapse levels across layers for ten state-of-the-art MoE-based LLMs. The results demonstrate that while all models exhibit router collapse, the intensity varies between reasoning and non-reasoning models. Best viewed in color.

1. Problems

Routers are a **key component** in Sparse Mixture of Experts (SMoE), but they remain challenging:

- **Difficult to learn:** Router optimization is challenging because routing decisions receive only **indirect** and **sparse gradient signals**.
- **Router collapse:** Our analysis of 10 **well-trained** large-scale MoEs ranging from 3B to 120B parameters shows severe collapse, with **collapse levels** between **0.4** and **0.9**

Is it possible to perform routing in MoEs without a router?

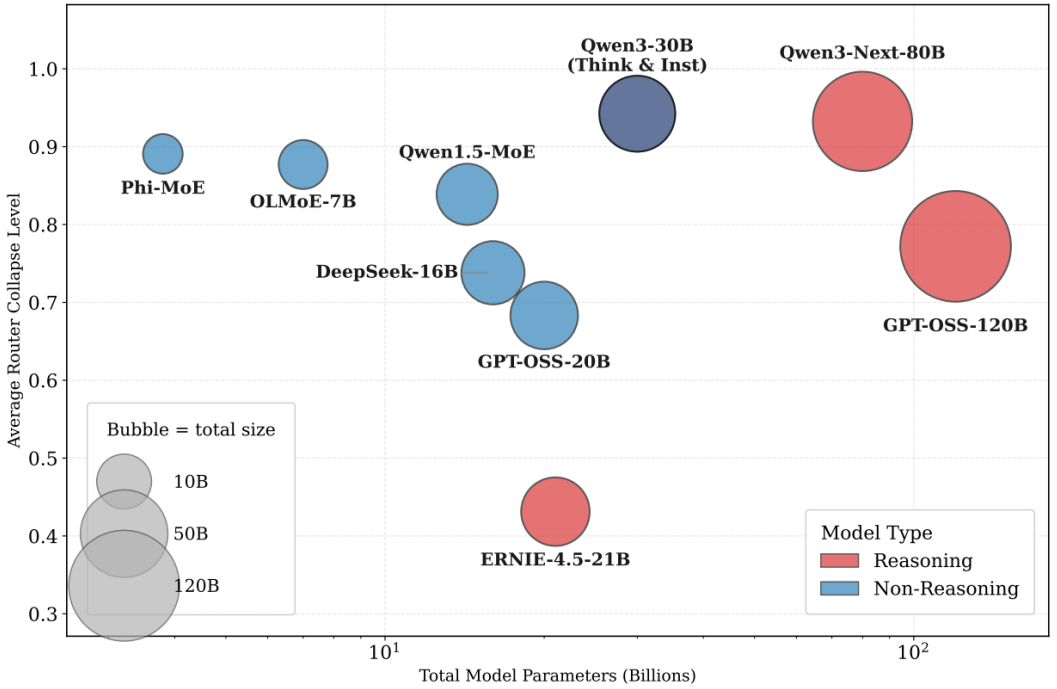


Figure 1. Average router collapse levels across layers for ten state-of-the-art MoE-based LLMs. The results demonstrate that while all models exhibit router collapse, the intensity varies between reasoning and non-reasoning models. Best viewed in color.

2. Findings

We study three well-trained SMoE-based LLMs — OLMoE, DeepSeekMoE, and QwenMoE — and find that:

- **Expert eigenvectors are effective representations:** *Eigenvectors* of expert weight matrices can be directly used for *downstream tasks*, including classification, clustering, reranking, STS and summarization.

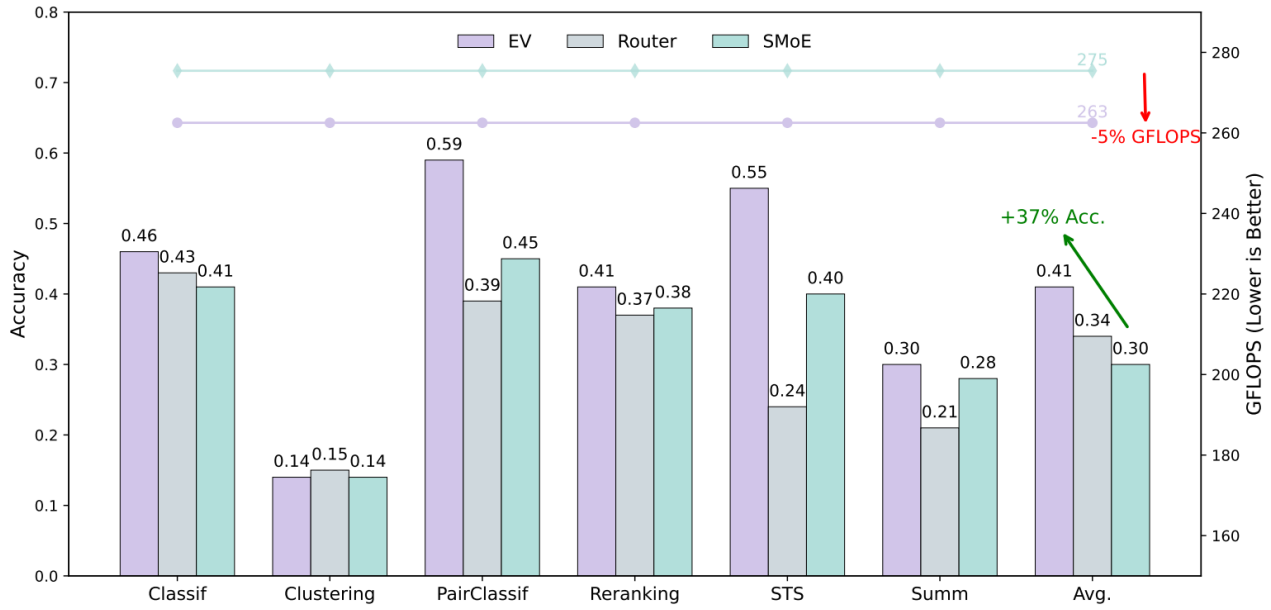


Figure 4. We compare the performance of the Eigenvector representation (EV) with traditional routers and *OLMoE-7B* (Muennighoff et al., 2025) on the Massive Text Embedding Benchmark (MTEB). The results show that EV outperforms OLMoE-7B on 37% of the tasks across six benchmarks while reducing computational cost by 5% in GFLOPs. These findings suggest that EV captures rich semantic information. Best viewed in color.

2. Findings

We study three well-trained SMoE-based LLMs — OLMoE, DeepSeekMoE, and QwenMoE — and find that:

- **Expert eigenvectors are effective representations:** *Eigenvectors* of expert weight matrices can be directly used for *downstream tasks*, including classification, clustering, reranking, STS and summarization.

- **Expert eigenvectors encode semantics:** These eigenvectors *capture* rich *semantic information*, revealing meaningful structure within trained experts.

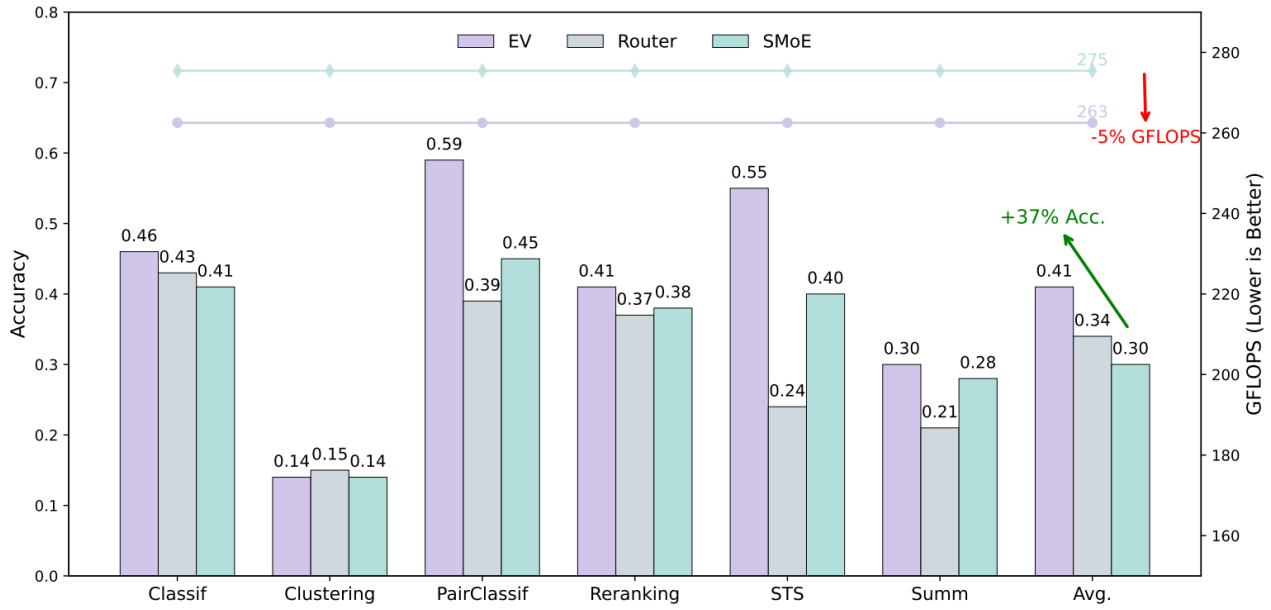


Figure 4. We compare the performance of the Eigenvector representation (EV) with traditional routers and *OLMoE-7B* (Muennighoff et al., 2025) on the Massive Text Embedding Benchmark (MTEB). The results show that EV outperforms OLMoE-7B on 37% of the tasks across six benchmarks while reducing computational cost by 5% in GFLOPs. These findings suggest that EV captures rich semantic information. Best viewed in color.

2. Findings

We study three well-trained SMoE-based LLMs — OLMoE, DeepSeekMoE, and QwenMoE — and find that:

- **Expert eigenvectors are effective representations:** *Eigenvectors* of expert weight matrices can be directly used for *downstream tasks*, including classification, clustering, reranking, STS and summarization.

- **Expert eigenvectors encode semantics:** These eigenvectors *capture* rich *semantic information*, revealing meaningful structure within trained experts.

Can we use expert eigenvectors as training-free routers for expert selection?

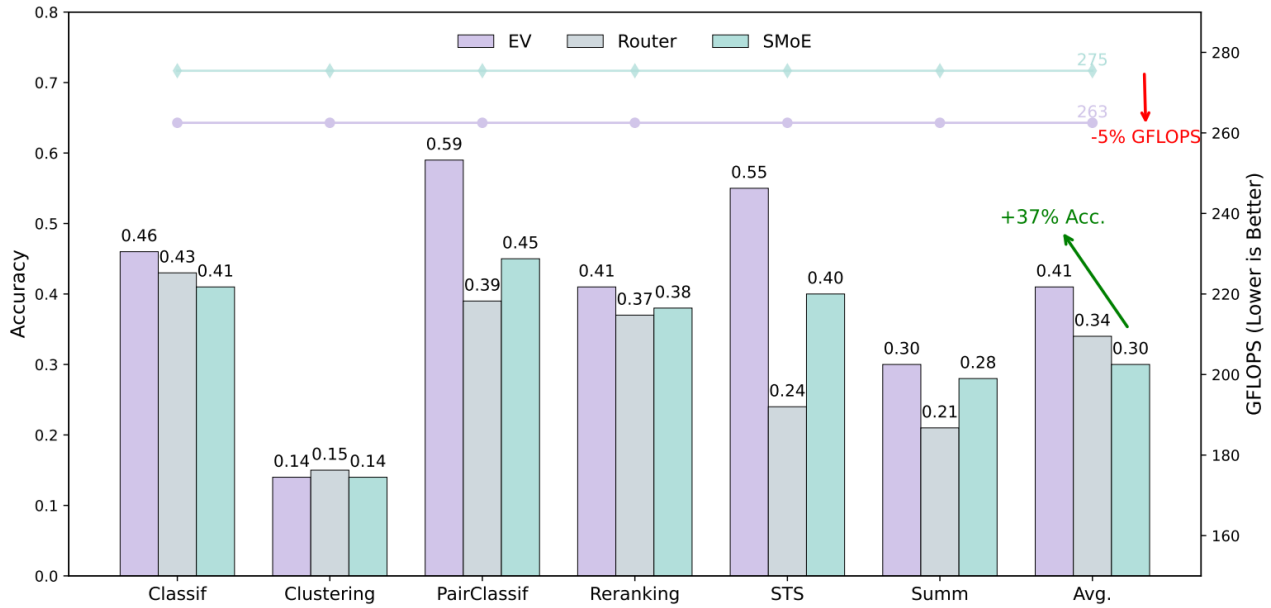


Figure 4. We compare the performance of the Eigenvector representation (EV) with traditional routers and *OLMoE-7B* (Muennighoff et al., 2025) on the Massive Text Embedding Benchmark (MTEB). The results show that EV outperforms OLMoE-7B on 37% of the tasks across six benchmarks while reducing computational cost by 5% in GFLOPs. These findings suggest that EV captures rich semantic information. Best viewed in color.

3. Solution: Eigenvectors Representation

Eigenvectors of Experts as Routers

Leverage the *spectral structure* of expert *weight* matrices to guide expert selection:

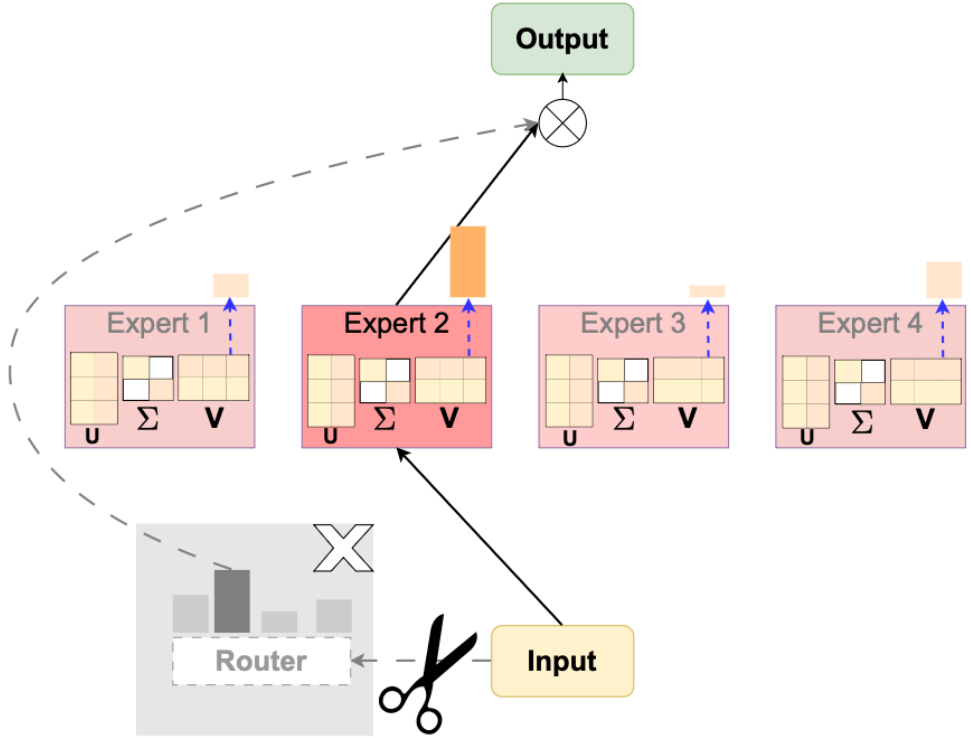


Figure 3. An illustration of our *Eigenvectors Representation*, which leverages enriched information from the eigenvectors of expert weights. In contrast, SMoE employs a learnable expert embedding (router) to select the top-k experts for each token. SSMoE provides an efficient and robust representation, as demonstrated in Section 4. Best viewed in color.

3. Solution: Eigenvectors Representation

Eigenvectors of Experts as Routers

Leverage the *spectral structure* of expert *weight* matrices to guide expert selection:

- **Training-free routing:** No additional router learning is required.

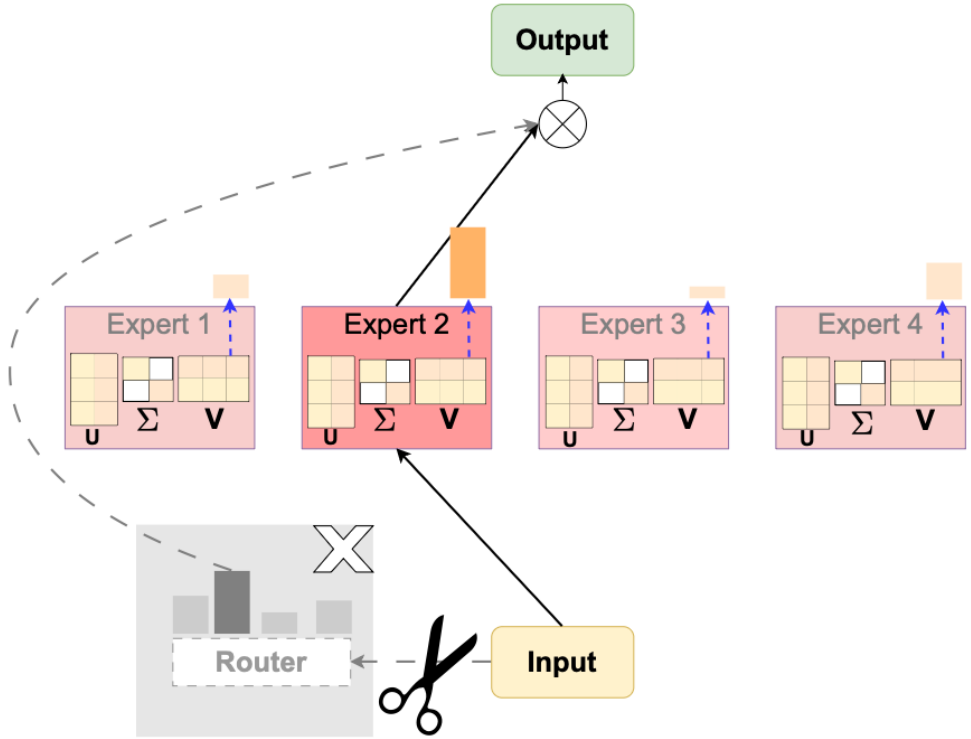


Figure 3. An illustration of our *Eigenvectors Representation*, which leverages enriched information from the eigenvectors of expert weights. In contrast, SMoE employs a learnable expert embedding (router) to select the top-k experts for each token. SSMoE provides an efficient and robust representation, as demonstrated in Section 4. Best viewed in color.

3. Solution: Eigenvectors Representation

Eigenvectors of Experts as Routers

Leverage the *spectral structure* of expert *weight* matrices to guide expert selection:

- **Training-free routing:** No additional router learning is required.
- **Non-collapsing expert selection:** Expert eigenvectors provide stable and diverse routing signals.

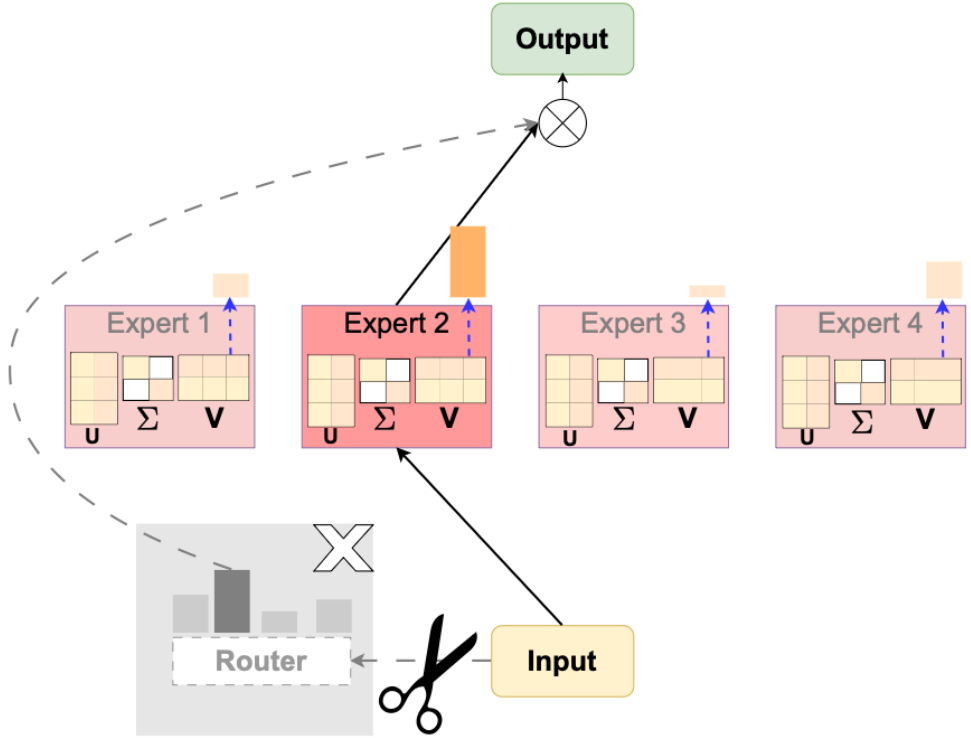


Figure 3. An illustration of our *Eigenvectors Representation*, which leverages enriched information from the eigenvectors of expert weights. In contrast, SMoE employs a learnable expert embedding (router) to select the top-k experts for each token. SSMoE provides an efficient and robust representation, as demonstrated in Section 4. Best viewed in color.

3. Solution: Singular Value Decomposition

SMoE (SSMoE)

SSMoE consists of 4 steps:

Step 1: Eigenvector Extraction. For each expert i, compute:

$$A^{(i)} = \mathrm{FFN}_1^{(i)} \mathrm{FFN}_1^{(i)\top} \in \mathbb{R}^{d\times d},$$
$$B^{(i)} = \mathrm{FFN}_2^{(i)\top} \mathrm{FFN}_2^{(i)} \in \mathbb{R}^{d\times d}. \quad (3)$$

Let $\{\boldsymbol{v}_j^{(i,1)}\}$ and $\{\boldsymbol{v}_j^{(i,2)}\}$ denote the eigenvectors of $A^{(i)}$ and $B^{(i)}$, respectively.

3. Solution: Singular Value Decomposition

SMoE (SSMoE)

SSMoE consists of 4 steps:

Step 1: Eigenvector Extraction. For each expert i, compute:

$$A^{(i)} = \mathrm{FFN}_1^{(i)}\,\mathrm{FFN}_1^{(i)\top} \in \mathbb{R}^{d\times d},$$
$$B^{(i)} = \mathrm{FFN}_2^{(i)\top}\,\mathrm{FFN}_2^{(i)} \in \mathbb{R}^{d\times d}. \tag{3}$$

Let $\{\boldsymbol{v}_j^{(i,1)}\}$ and $\{\boldsymbol{v}_j^{(i,2)}\}$ denote the eigenvectors of $A^{(i)}$ and $B^{(i)}$, respectively.

Step 2: Top-c Eigenvector Selection. Given the router embedding $\boldsymbol{r}^{(i)} \in \mathbb{R}^d$, select the top-$c$ most similar eigenvectors from both sets:

$$\mathcal{C}_1^{(i)} = \mathrm{TopC}\big(\{\boldsymbol{v}_j^{(i,1)}\},\, \boldsymbol{r}^{(i)},\, c\big),$$
$$\mathcal{C}_2^{(i)} = \mathrm{TopC}\big(\{\boldsymbol{v}_j^{(i,2)}\},\, \boldsymbol{r}^{(i)},\, c\big). \tag{4}$$

Then compute the average to obtain the spectral embedding for expert i:

$$\mathcal{C}^{(i)} = \frac{1}{2}\left(\mathrm{mean}(\mathcal{C}_1^{(i)}) + \mathrm{mean}(\mathcal{C}_2^{(i)})\right) \in \mathbb{R}^d$$

3. Solution: Singular Value Decomposition

SMoE (SSMoE)

SSMoE consists of 4 steps:

Step 1: Eigenvector Extraction. For each expert i, compute:

$$A^{(i)} = \mathrm{FFN}_1^{(i)}\,\mathrm{FFN}_1^{(i)\top} \in \mathbb{R}^{d\times d},$$
$$B^{(i)} = \mathrm{FFN}_2^{(i)\top}\,\mathrm{FFN}_2^{(i)} \in \mathbb{R}^{d\times d}. \tag{3}$$

Let $\{\boldsymbol{v}_j^{(i,1)}\}$ and $\{\boldsymbol{v}_j^{(i,2)}\}$ denote the eigenvectors of $A^{(i)}$ and $B^{(i)}$, respectively.

Step 2: Top-c Eigenvector Selection. Given the router embedding $\boldsymbol{r}^{(i)} \in \mathbb{R}^d$, select the top-$c$ most similar eigenvectors from both sets:

$$\mathcal{C}_1^{(i)} = \mathrm{TopC}\big(\{\boldsymbol{v}_j^{(i,1)}\},\, \boldsymbol{r}^{(i)},\, c\big),$$
$$\mathcal{C}_2^{(i)} = \mathrm{TopC}\big(\{\boldsymbol{v}_j^{(i,2)}\},\, \boldsymbol{r}^{(i)},\, c\big). \tag{4}$$

Then compute the average to obtain the spectral embedding for expert i:

$$\mathcal{C}^{(i)} = \frac{1}{2}\left(\mathrm{mean}(\mathcal{C}_1^{(i)}) + \mathrm{mean}(\mathcal{C}_2^{(i)})\right) \in \mathbb{R}^d$$

Step 3: Spectral Routing Matrix. Concatenate the expert spectral embeddings to form the routing weight matrix:

$$W_{\mathrm{EV}} = \left[\mathcal{C}^{(1)}\ \mathcal{C}^{(2)}\ \ldots\ \mathcal{C}^{(N)}\right] \in \mathbb{R}^{d\times N}$$

3. Solution: Singular Value Decomposition

SMoE (SSMoE)

SSMoE consists of 4 steps:

Step 1: Eigenvector Extraction. For each expert i, compute:

$$A^{(i)} = \mathrm{FFN}_1^{(i)}\,\mathrm{FFN}_1^{(i)\top} \in \mathbb{R}^{d\times d},$$
$$B^{(i)} = \mathrm{FFN}_2^{(i)\top}\,\mathrm{FFN}_2^{(i)} \in \mathbb{R}^{d\times d}. \tag{3}$$

Let $\{\boldsymbol{v}_j^{(i,1)}\}$ and $\{\boldsymbol{v}_j^{(i,2)}\}$ denote the eigenvectors of $A^{(i)}$ and $B^{(i)}$, respectively.

Step 2: Top-c Eigenvector Selection. Given the router embedding $\boldsymbol{r}^{(i)} \in \mathbb{R}^d$, select the top-$c$ most similar eigenvectors from both sets:

$$\mathcal{C}_1^{(i)} = \mathrm{TopC}\big(\{\boldsymbol{v}_j^{(i,1)}\},\, \boldsymbol{r}^{(i)},\, c\big),$$
$$\mathcal{C}_2^{(i)} = \mathrm{TopC}\big(\{\boldsymbol{v}_j^{(i,2)}\},\, \boldsymbol{r}^{(i)},\, c\big). \tag{4}$$

Then compute the average to obtain the spectral embedding for expert i:

$$\mathcal{C}^{(i)} = \frac{1}{2}\left(\mathrm{mean}(\mathcal{C}_1^{(i)}) + \mathrm{mean}(\mathcal{C}_2^{(i)})\right) \in \mathbb{R}^d$$

Step 3: Spectral Routing Matrix. Concatenate the expert spectral embeddings to form the routing weight matrix:

$$W_{\mathrm{EV}} = \left[\mathcal{C}^{(1)}\ \mathcal{C}^{(2)}\ \ldots\ \mathcal{C}^{(N)}\right] \in \mathbb{R}^{d\times N}$$

Step 4: Eigenvectors Representation.

$$f_{\mathrm{EV}}(\boldsymbol{x}) = \boldsymbol{x}W_{\mathrm{EV}} \in \mathbb{R}^{b\times N}$$

$$f_{\mathrm{SSMoE}}(\boldsymbol{x}) = \sum_{i\in \mathrm{TopK}(\mathcal{V}(\boldsymbol{x}),k)} \mathcal{V}_i(\boldsymbol{x})\cdot E_i(\boldsymbol{x}), \tag{5}$$

where $\mathcal{V}(\boldsymbol{x}) = \mathrm{softmax}(f_{\mathrm{EV}}(\boldsymbol{x}))\cdot\alpha + \mathcal{S}(\boldsymbol{x})\cdot(1-\alpha)$, and $\alpha \in [0,1]$ is a balancing factor that interpolates between the EV Router and the standard SMoE Router.

4. Results

Theoretical Insights of SSMoE

- **Approximate expert orthogonality (Lemma B.1)**: Expert eigenvectors are approximately orthogonal, providing diverse routing directions.
- **Collapse mitigation (Lemma 3.1):** The SSMoE router reduces representation collapse, a common issue in conventional SMoE models (Chi et al., 2022).

4. Results

Theoretical Insights of SSMoE

- **Approximate expert orthogonality (Lemma B.1)**: Expert eigenvectors are approximately orthogonal, providing diverse routing directions.
- **Collapse mitigation (Lemma 3.1):** The SSMoE router reduces representation collapse, a common issue in conventional SMoE models (Chi et al., 2022).

4. Results

- **Full-expert setting:** SSMoE *improves* reasoning performance over the original model *without retraining*.

Model	Method	Memory (GB)	ARC-C	ARC-E	BoolQ	GSM8K	HellaSwag	OBQA	PIQA	WinoGrande	Avg.
GPT-OSS-20B 10-shot	Original	38.96	$44.2_{\pm 1.5}$	$77.9_{\pm 0.9}$	$62.8_{\pm 0.8}$	$\mathbf{36.8}_{\pm 1.3}$	$29.0_{\pm 0.5}$	$19.8_{\pm 1.8}$	$62.5_{\pm 1.1}$	$57.5_{\pm 1.4}$	$48.8_{\pm 1.2}$
	SSMoE-FULL (Ours)	38.96	$\mathbf{50.2}_{\pm 1.5}$	$\mathbf{79.3}_{\pm 0.8}$	$71.7_{\pm 0.8}$	$43.4_{\pm 1.4}$	$\mathbf{40.9}_{\pm 0.5}$	$\mathbf{33.8}_{\pm 2.1}$	$\mathbf{63.9}_{\pm 1.1}$	$\mathbf{58.3}_{\pm 1.4}$	$\mathbf{55.2}_{\pm 1.1}$
	RandDrop (Baseline)	30.05	$33.5_{\pm 1.4}$	$65.8_{\pm 1.0}$	$66.5_{\pm 0.8}$	$33.1_{\pm 1.3}$	$29.7_{\pm 0.5}$	$20.6_{\pm 1.8}$	$57.2_{\pm 1.2}$	$53.3_{\pm 1.4}$	$44.9_{\pm 1.2}$
	Router	30.05	$48.6_{\pm 1.5}$	$80.1_{\pm 0.8}$	$\mathbf{76.6}_{\pm 0.7}$	$22.3_{\pm 1.1}$	$35.6_{\pm 0.5}$	$24.2_{\pm 1.9}$	$71.3_{\pm 1.1}$	$60.3_{\pm 1.4}$	$52.4_{\pm 1.1}$
	SSMoE (Ours)	30.05	$\mathbf{52.1}_{\pm 1.5}$	$\mathbf{82.4}_{\pm 0.8}$	$72.4_{\pm 0.8}$	$30.8_{\pm 1.3}$	$\mathbf{37.5}_{\pm 0.5}$	$\mathbf{23.0}_{\pm 1.9}$	$\mathbf{72.2}_{\pm 1.0}$	$\mathbf{62.2}_{\pm 1.4}$	$\mathbf{54.1}_{\pm 1.1}$
	vs. Baseline	0.0%	+55.5%	+25.2%	+8.9%	-6.9%	+26.3%	+11.7%	+26.2%	+16.7%	+20.5%
	vs. Original	-22.9%	+17.9%	+5.8%	+15.3%	-16.3%	+29.3%	+16.2%	+15.5%	+8.2%	+10.9%
GPT-OSS-120B 10-shot	Original	217.61	$\mathbf{52.4}_{\pm 1.5}$	$\mathbf{83.0}_{\pm 0.8}$	$64.0_{\pm 0.8}$	$49.3_{\pm 1.4}$	$30.5_{\pm 0.5}$	$22.0_{\pm 1.9}$	$62.9_{\pm 1.1}$	$56.7_{\pm 1.4}$	$52.6_{\pm 1.2}$
	RandDrop (Baseline)	164.20	$43.3_{\pm 1.4}$	$74.1_{\pm 0.9}$	$74.2_{\pm 0.8}$	$46.7_{\pm 1.4}$	$33.2_{\pm 0.5}$	$\mathbf{21.4}_{\pm 1.8}$	$56.0_{\pm 1.2}$	$51.8_{\pm 1.4}$	$50.1_{\pm 1.2}$
	Router	164.20	$42.4_{\pm 1.4}$	$76.1_{\pm 0.9}$	$77.6_{\pm 0.7}$	$49.2_{\pm 1.4}$	$36.2_{\pm 0.5}$	$20.6_{\pm 1.8}$	$\mathbf{65.1}_{\pm 1.1}$	$\mathbf{58.2}_{\pm 1.4}$	$53.2_{\pm 1.2}$
	SSMoE (Ours)	164.20	$45.6_{\pm 1.5}$	$79.1_{\pm 0.8}$	$\mathbf{79.3}_{\pm 0.7}$	$\mathbf{57.6}_{\pm 1.4}$	$\mathbf{37.1}_{\pm 0.5}$	$21.2_{\pm 1.8}$	$63.3_{\pm 1.1}$	$56.4_{\pm 1.4}$	$\mathbf{54.9}_{\pm 1.1}$
	vs. Baseline	0.0%	+5.3%	+6.8%	+6.9%	+23.3%	+11.7%	-0.9%	+13.0%	+8.9%	+9.6%
	vs. Original	-24.5%	-13.0%	-4.7%	+23.9%	+16.8%	-3.6%	+0.6%	-0.5%	+4.4%	+4.4%

Table 1. Performance comparison of different methods on GPT-OSS models with 10-shot evaluation. Results show mean ± standard deviation across eight benchmarks. SSMoE-FULL denotes our SSMoE variant using 100% of experts, enabling a memory-matched comparison with the original model. Best results in each model size are shown in **bold**. Improvements are calculated relative to baseline methods.

4. Results

- **Full-expert setting:** SSMoE *improves* reasoning performance over the original model *without retraining*.
- **Expert-pruned setting:** SSMoE outperforms the original model while activating only *75% of experts*.

Model	Method	Memory (GB)	ARC-C	ARC-E	BoolQ	GSM8K	HellaSwag	OBQA	PIQA	WinoGrande	Avg.
GPT-OSS-20B 10-shot	Original	38.96	$44.2_{\pm 1.5}$	$77.9_{\pm 0.9}$	$62.8_{\pm 0.8}$	$\mathbf{36.8}_{\pm 1.3}$	$29.0_{\pm 0.5}$	$19.8_{\pm 1.8}$	$62.5_{\pm 1.1}$	$57.5_{\pm 1.4}$	$48.8_{\pm 1.2}$
	SSMoE-FULL (Ours)	38.96	$\mathbf{50.2}_{\pm 1.5}$	$\mathbf{79.3}_{\pm 0.8}$	$71.7_{\pm 0.8}$	$43.4_{\pm 1.4}$	$\mathbf{40.9}_{\pm 0.5}$	$\mathbf{33.8}_{\pm 2.1}$	$\mathbf{63.9}_{\pm 1.1}$	$\mathbf{58.3}_{\pm 1.4}$	$\mathbf{55.2}_{\pm 1.1}$
	RandDrop (Baseline)	30.05	$33.5_{\pm 1.4}$	$65.8_{\pm 1.0}$	$66.5_{\pm 0.8}$	$33.1_{\pm 1.3}$	$29.7_{\pm 0.5}$	$20.6_{\pm 1.8}$	$57.2_{\pm 1.2}$	$53.3_{\pm 1.4}$	$44.9_{\pm 1.2}$
	Router	30.05	$48.6_{\pm 1.5}$	$80.1_{\pm 0.8}$	$\mathbf{76.6}_{\pm 0.7}$	$22.3_{\pm 1.1}$	$35.6_{\pm 0.5}$	$24.2_{\pm 1.9}$	$71.3_{\pm 1.1}$	$60.3_{\pm 1.4}$	$52.4_{\pm 1.1}$
	SSMoE (Ours)	30.05	$\mathbf{52.1}_{\pm 1.5}$	$\mathbf{82.4}_{\pm 0.8}$	$72.4_{\pm 0.8}$	$30.8_{\pm 1.3}$	$\mathbf{37.5}_{\pm 0.5}$	$\mathbf{23.0}_{\pm 1.9}$	$\mathbf{72.2}_{\pm 1.0}$	$\mathbf{62.2}_{\pm 1.4}$	$\mathbf{54.1}_{\pm 1.1}$
	vs. Baseline	0.0%	+55.5%	+25.2%	+8.9%	-6.9%	+26.3%	+11.7%	+26.2%	+16.7%	+20.5%
	vs. Original	-22.9%	+17.9%	+5.8%	+15.3%	-16.3%	+29.3%	+16.2%	+15.5%	+8.2%	+10.9%
GPT-OSS-120B 10-shot	Original	217.61	$\mathbf{52.4}_{\pm 1.5}$	$\mathbf{83.0}_{\pm 0.8}$	$64.0_{\pm 0.8}$	$49.3_{\pm 1.4}$	$30.5_{\pm 0.5}$	$22.0_{\pm 1.9}$	$62.9_{\pm 1.1}$	$56.7_{\pm 1.4}$	$52.6_{\pm 1.2}$
	RandDrop (Baseline)	164.20	$43.3_{\pm 1.4}$	$74.1_{\pm 0.9}$	$74.2_{\pm 0.8}$	$46.7_{\pm 1.4}$	$33.2_{\pm 0.5}$	$\mathbf{21.4}_{\pm 1.8}$	$56.0_{\pm 1.2}$	$51.8_{\pm 1.4}$	$50.1_{\pm 1.2}$
	Router	164.20	$42.4_{\pm 1.4}$	$76.1_{\pm 0.9}$	$77.6_{\pm 0.7}$	$49.2_{\pm 1.4}$	$36.2_{\pm 0.5}$	$20.6_{\pm 1.8}$	$\mathbf{65.1}_{\pm 1.1}$	$\mathbf{58.2}_{\pm 1.4}$	$53.2_{\pm 1.2}$
	SSMoE (Ours)	164.20	$45.6_{\pm 1.5}$	$79.1_{\pm 0.8}$	$\mathbf{79.3}_{\pm 0.7}$	$\mathbf{57.6}_{\pm 1.4}$	$\mathbf{37.1}_{\pm 0.5}$	$21.2_{\pm 1.8}$	$63.3_{\pm 1.1}$	$56.4_{\pm 1.4}$	$\mathbf{54.9}_{\pm 1.1}$
	vs. Baseline	0.0%	+5.3%	+6.8%	+6.9%	+23.3%	+11.7%	-0.9%	+13.0%	+8.9%	+9.6%
	vs. Original	-24.5%	-13.0%	-4.7%	+23.9%	+16.8%	-3.6%	+0.6%	-0.5%	+4.4%	+4.4%

Table 1. Performance comparison of different methods on GPT-OSS models with 10-shot evaluation. Results show mean ± standard deviation across eight benchmarks. SSMoE-FULL denotes our SSMoE variant using 100% of experts, enabling a memory-matched comparison with the original model. Best results in each model size are shown in **bold**. Improvements are calculated relative to baseline methods.

4. Results

- **Semantic Evaluation:** SSMoE consistently improves over baselines across advanced SMoE models, including OLMoE-7B and DeepSeekMoE-16B, *with 25–30% gains*.
- **Robust Evaluation:** SSMoE reaches SOTA performance on *15/18 clean tasks* and *16/18 corrupt tasks*, showing strong robustness under distribution shift.

		Clean					Corrupt				
Model	**Task**	**Router**	**SMoE**	**MoEE**	**EV**	**SSMoE**	**Router**	**SMoE**	**MoEE**	**EV**	**SSMoE**
OLMoE-7B	Classification	41.2	43.4	41.8	45.9	**50.6**	36.0	36.9	36.1	36.3	**39.7**
	Clustering	13.7	14.7	14.5	14.2	**16.4**	6.7	6.8	6.8	6.9	**7.1**
	PairClassification	45.3	39.1	45.7	**59.2**	51.2	28.1	26.7	27.6	29.1	**30.7**
	Reranking	37.5	37.4	39.5	41.2	**42.4**	27.8	26.8	27.7	28.2	**29.3**
	STS	39.9	24.1	39.9	**55.3**	47.0	5.1	1.9	3.6	7.5	**13.4**
	Summarization	28.4	20.9	29.8	29.6	**30.6**	24.2	24.5	25.3	26.4	**28.5**
	Avg.	34.3	29.9	35.2	**40.9**	39.7	21.3	20.6	21.2	22.4	**24.8**
Qwen-MoE-7B	Classification	43.8	50.3	47.7	43.4	**51.6**	34.4	38.3	37.0	36.9	**39.8**
	Clustering	13.6	27.4	25.2	13.7	**31.1**	5.8	8.2	7.2	6.1	**8.6**
	PairClassification	45.9	46.9	51.5	47.0	**55.4**	26.6	27.4	30.1	**31.7**	30.7
	Reranking	39.6	45.3	48.5	40.3	**49.1**	26.2	28.9	29.4	26.7	**29.8**
	STS	38.8	38.0	51.8	41.6	**57.9**	0.7	0.7	2.0	1.7	**6.8**
	Summarization	28.3	13.4	**31.2**	30.3	29.6	26.7	23.0	25.5	27.9	**28.5**
	Avg.	35.0	36.9	42.6	36.0	**45.8**	20.1	21.1	21.9	21.8	**24.0**
DeepSeekMoE-16B	Classification	43.4	46.6	44.4	44.0	**48.6**	36.2	36.7	36.0	**40.0**	39.7
	Clustering	13.4	18.1	17.8	14.3	**21.6**	6.6	6.8	6.8	7.2	**8.3**
	PairClassification	45.5	40.9	46.1	46.7	**51.2**	26.9	25.9	27.3	28.0	**31.5**
	Reranking	38.5	38.9	42.2	39.2	**44.9**	38.8	38.3	39.0	39.5	**41.0**
	STS	37.7	26.3	40.2	40.0	**50.1**	5.1	3.7	5.7	9.1	**14.3**
	Summarization	24.9	22.0	24.4	28.9	**29.9**	27.8	25.2	28.3	28.3	**28.4**
	Avg.	33.9	32.1	35.9	35.5	**41.1**	23.6	22.8	23.8	25.4	**27.2**

Table 2. Performance comparison across MTEB tasks under **clean** and **corrupt** settings. Results are shown side-by-side. The best score in each row (clean or corrupt) is highlighted in **bold**.

4. Results

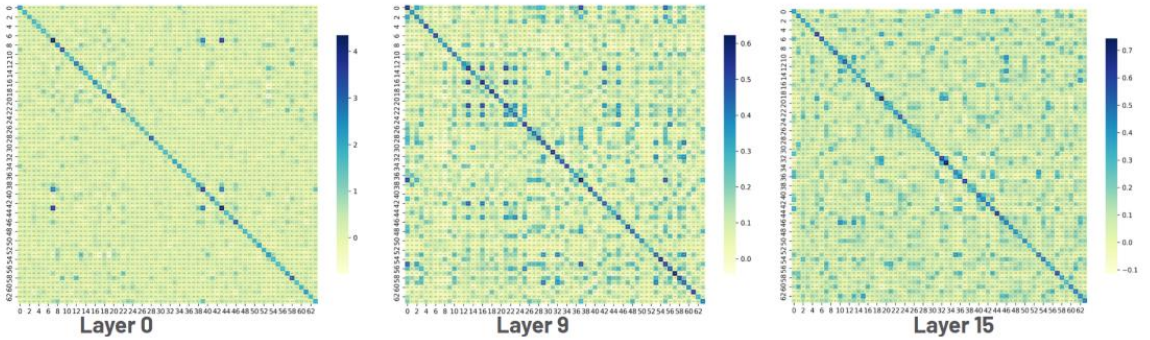
- **Image-Text Retrieval Evaluation:** SSMoE *outperforms* both the original CLIP model and the CLIP-MoE baseline across all *retrieval tasks*, under both clean and corrupted settings.
- **Statistical Significance Test:** We report the original baseline score, SSMoE score, absolute improvement, *95% confidence interval*, and paired bootstrap p-value to validate the significance of SSMoE's gains.

Setting	Model	COCO I2T			COCO T2I		
		@1	@5	@10	@1	@5	@10
	CLIP (OpenAI)	56.1	79.5	86.8	35.4	60.1	70.2
Clean	CLIP-MoE	65.0	86.0	92.0	46.8	71.7	80.4
	SSMoE (Ours)	**65.7**	**86.5**	**92.3**	**47.1**	**72.0**	**80.7**
	% Improv.	**+1.1%**	**+0.6%**	**+0.3%**	**+0.6%**	**+0.4%**	**+0.4%**
Corrupt	CLIP-MoE	36.2	60.2	69.9	29.4	51.7	61.7
	SSMoE (Ours)	**37.2**	**61.0**	**70.3**	**30.0**	**52.1**	**62.2**
	% Improv.	**+2.8%**	**+1.3%**	**+0.6%**	**+2.0%**	**+0.8%**	**+0.8%**

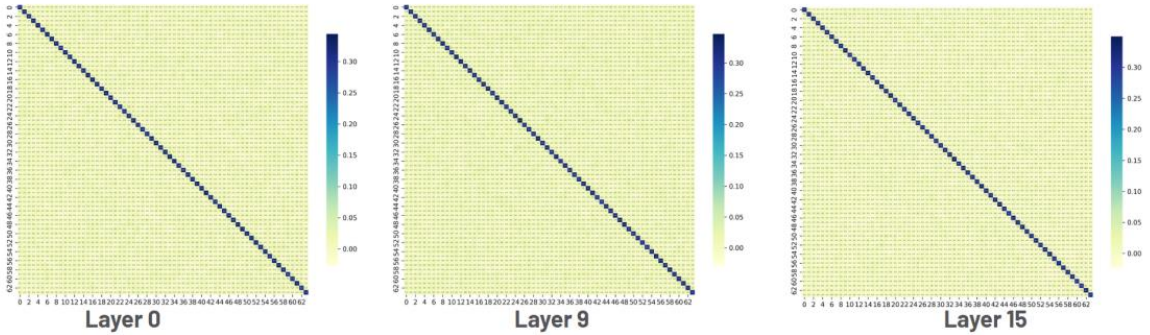
Metric	CLIP-MoE	SSMoE (Ours)	Δ	95% CI	p-value
COCO I2T					
@1	65.0	65.9	+0.8	[0.20, 1.58]	0.002
@5	86.0	86.5	+0.5	[0.00, 1.00]	0.024
@10	92.0	92.2	+0.2	[0.05, 0.40]	0.018
Flickr30k I2T					
@1	61.8	62.3	+0.5	[0.15, 0.80]	0.001
@5	83.4	84.1	+0.6	[0.40, 0.88]	< 0.001
@10	89.6	89.9	+0.3	[0.06, 0.44]	0.008

4. Results - Analysis

- **Original Router:** Expert representations are highly *correlated*, leading to *redundant* routing directions.
- **EV Router:** Expert eigenvectors are approximately *orthogonal*, providing *diverse* and *non-collapsing* routing directions.



(a) Correlation between expert embeddings within the SMoE router of the OLMoE model.



(b) Correlation between expert weight eigenvectors (EV router).

Figure 6. A comparison of collapse behavior shows that the OL-MoE router exhibits increased collapse at deeper layers, while the EV router does not encounter this issue.

4. Results - Analysis

- SSMoE leverages strong **semantic signals** from expert eigenvectors, producing representations with more *coherent* and *well-separated clusters*.

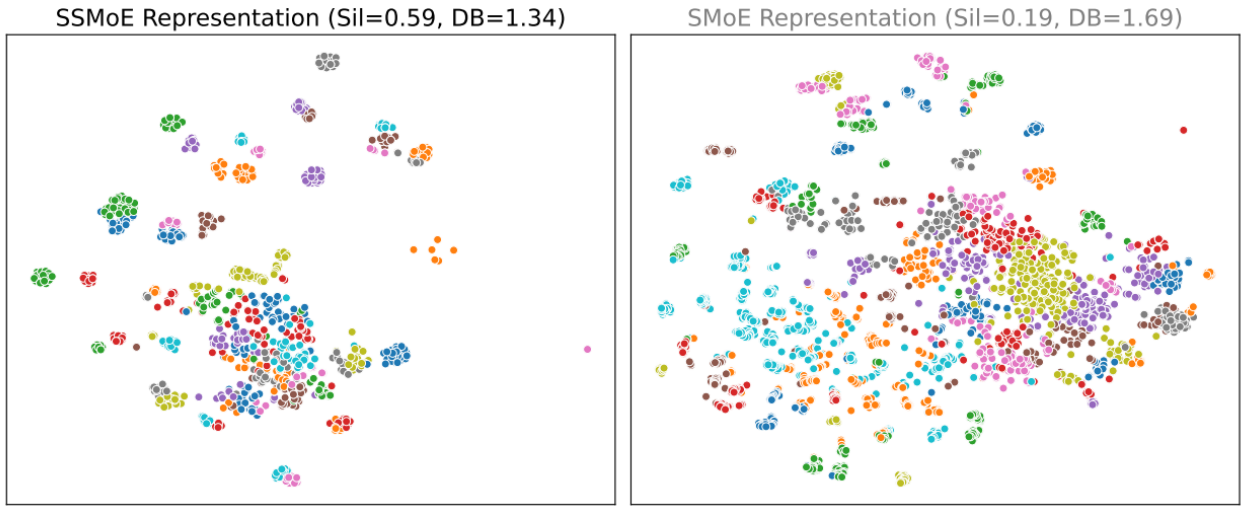


Figure 7. t-SNE visualizations of two learned representations, colored by KMeans cluster assignments with $k = 64$. SSMoE (left) displays more compact and well-separated clusters than SMoE (right), indicating stronger underlying cluster structure. Clustering quality is further supported by quantitative metrics: higher Silhouette and lower Davies-Bouldin index for SSMoE.

5. Conclusion

- **Conventional MoE routers are fragile:** Although routers are central to MoE models, they often suffer from severe *collapse* even in well-trained large-scale MoEs.
- **Expert eigenvectors encode semantics:** Eigenvectors of expert weight matrices contain rich *semantic information* that can guide expert selection.
- **SSMoE enables training-free routing:** SSMoE provides a *training-free* and *non-collapsing* router, mitigating representation collapse and improving model performance without retraining.
- **New direction for MoE design:** Our findings open a *new path* toward *redesigning* MoEs beyond conventional learned routers.

End