Enhancing Mixture-of-Experts Specialization via Cluster-Aware Upcycling

Sanghyeok Chu* Pyunghwan Ahn† Gwangmo Song Seung Hwan Kim Honglak Lee Bohyung Han†

Seoul National University · LG AI Research · University of Michigan

** Work done during an internship at LG AI Research. † Corresponding authors.*

Why MoE?

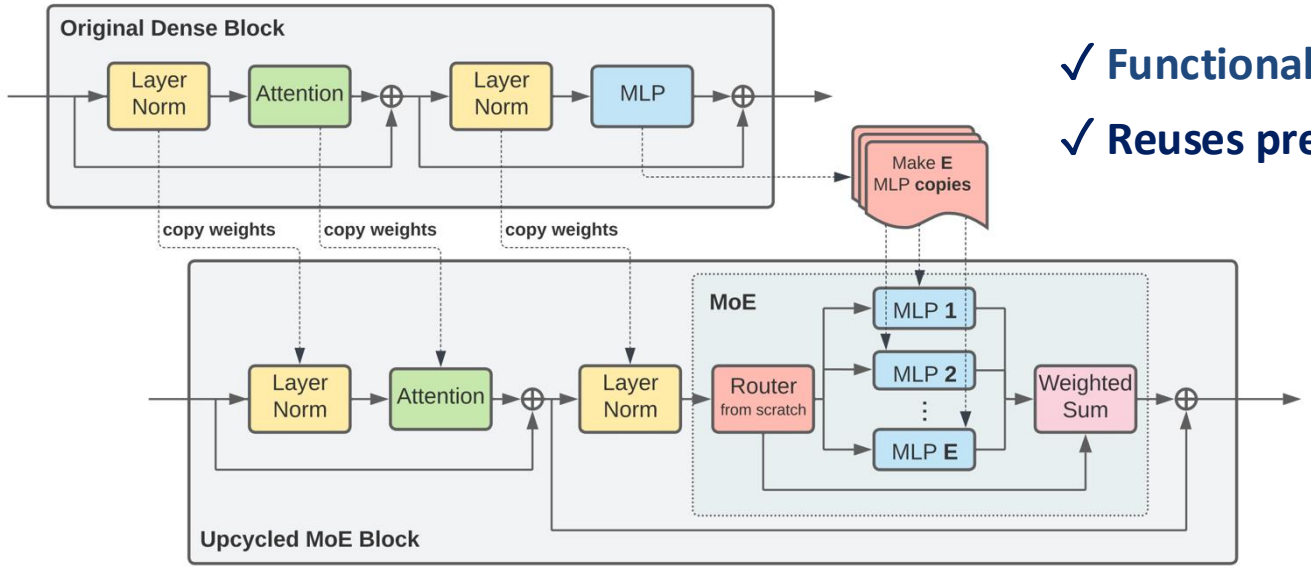
- Scaling model size reliably improves performance[1].
  - Performance ∝ 1) dataset size, 2) model size, and 3) total compute.
- But in dense architectures, training and inference costs grow proportionally to model size.
- Mixture-of-Experts (MoE): conditional computation
  - N_e experts per layer, with sparse top-k routing per token.
  - Total parameters ↑, but only a subset is activated per token.
    - Increases model capacity without proportional compute growth.
  - Especially beneficial at inference time.

However, training MoE models from scratch is prohibitively expensive ⇒ This motivates MoE upcycling.

[1] Kaplan et al., “Scaling Laws for Neural Language Models”. In arXiv, 2020.

Sparse Upcycling [Komatsuzaki et al., ICLR 2023]

- Start from a pretrained dense Transformer.
  - 1) Replace FFN layers with MoE layers, each consisting of N_e experts and a router.
  - 2) Initialize every experts by copying the dense FFN weights: $\forall i, \boldsymbol{W}_i \leftarrow \boldsymbol{W}$.
  - 3) Randomly initialize the router weights $\boldsymbol{W}_r$.



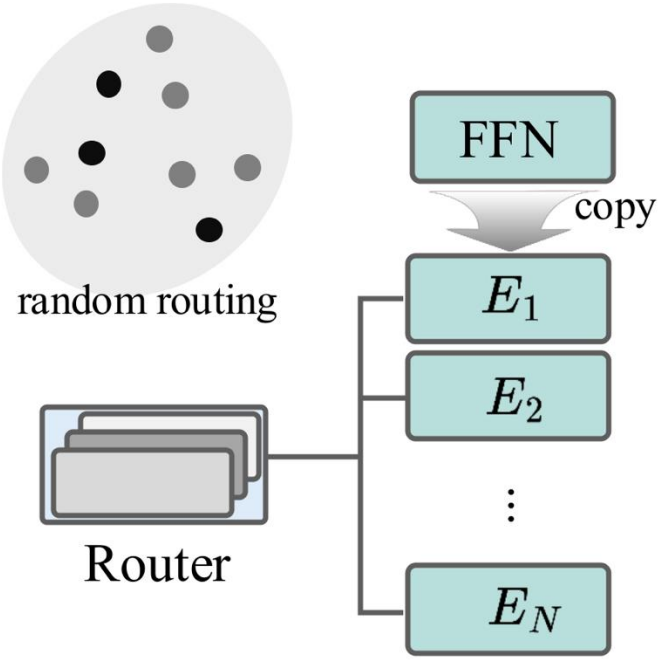
✓ **Functionally equivalent to the dense model at initialization.**

✓ **Reuses pretrained knowledge for faster convergence.**

[2] Komatsuzaki et al., "Sparse Upcycling: Training Mixture-of-Experts from Dense Checkpoints". In ICLR, 2023.

Problem: Expert Symmetry

- The experts are initially indistinguishable, making it challenging to establish a meaningful basis for specialized routing.
- Existing approaches attempt to break expert symmetry:
  - Noise injection: marginal performance gains.
  - Domain-specific fine-tuning: requires manually-defined domains and additional fine-tuning stages.
  - Drop-Upcycling[3]: partially reinitializes expert parameters, but disrupts the pretrained representation space.



[3] Nakamura et al., “Drop-Upcycling: Training Sparse Mixture of Experts with Partial Re-initialization”. In ICLR, 2025.

Our Idea: Cluster-aware Upcycling

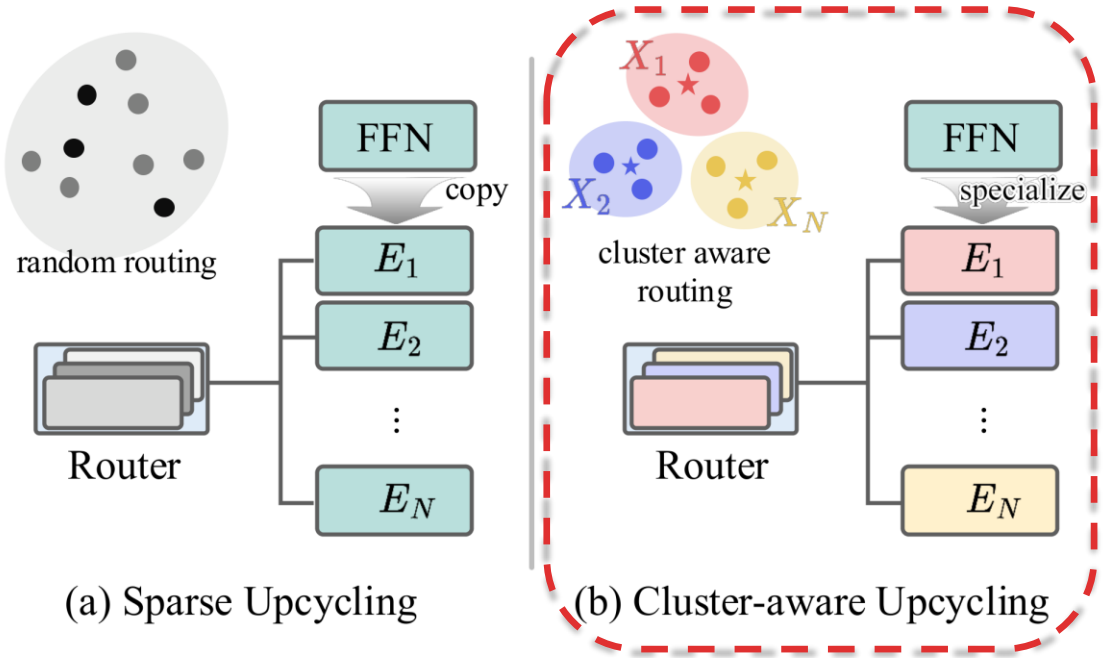
Use the **semantic structure** in pretrained dense activations to establish expert specialization at initialization, then **preserve and reinforce** it with ensemble-level self-distillation.

1. Cluster-aware Initialization

Use activation clusters as semantic priors for expert specialization and router assignment.

2. Expert-Ensemble Self-Distillation

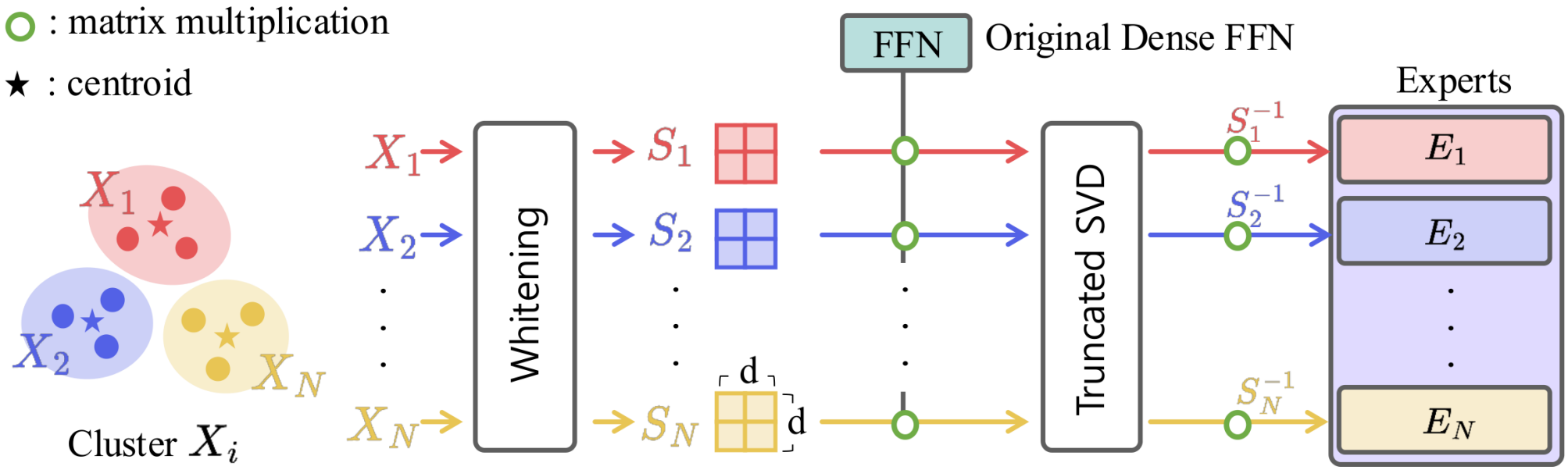
Use dense EMA ensemble-level supervision to preserve and reinforce expert specialization.



Cluster-aware Initialization

Cluster-aware Initialization consists of three steps:

1) Activation clustering
2) Cluster-aware expert initialization
3) Cluster-aware router initialization



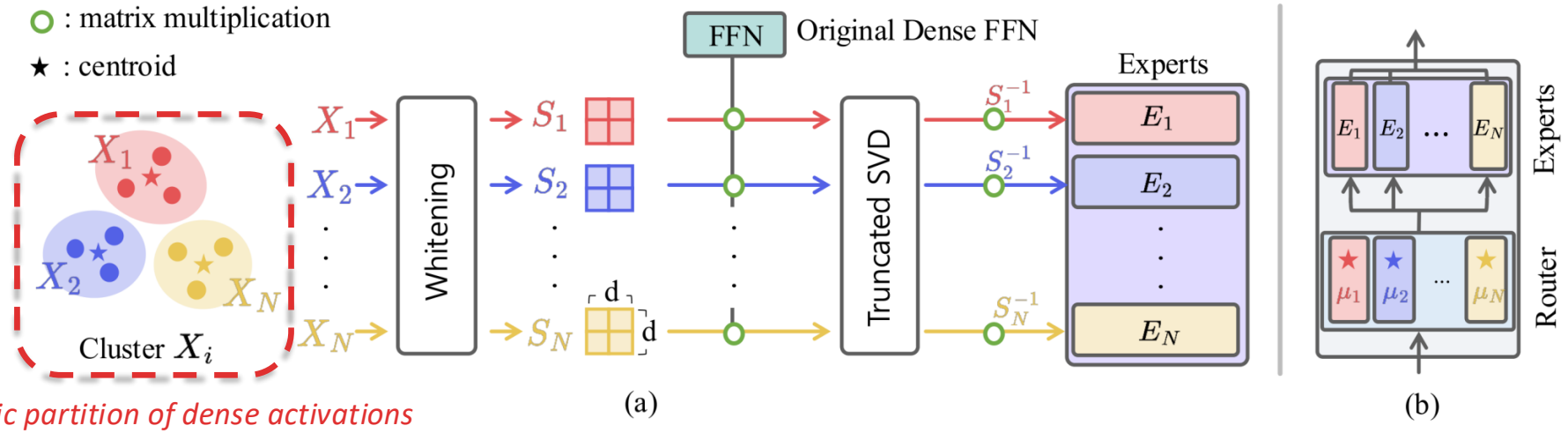
Step 1: Input Activation Clustering

Partition the dense model's activation space to extract a semantic prior for MoE initialization.

- Collect FFN input activations $\boldsymbol{X}$ from each FFN block.
- Apply spherical *k*-means based on cosine similarity.

$$\{\boldsymbol{\mu}_i\}_{i=1}^{N_e} = \underset{\{\hat{\boldsymbol{\mu}}_i : \|\hat{\boldsymbol{\mu}}_i\|_2 = 1\}_{i=1}^{N_e}}{\arg\max} \sum_{j=1}^{M} \max_i \hat{\boldsymbol{\mu}}_i^T \boldsymbol{x}_j$$

- Cosine similarity aligns with router logits $\boldsymbol{W}_r \boldsymbol{x}$, which measure directional alignment.



Semantic partition of dense activations

Step 2: Cluster-aware Expert Initialization

Objective:

1) Encourage each expert to approximate the dense model within its assigned cluster,
2) while discouraging redundancy across experts.

$$\min_{\{\boldsymbol{W}_i\}_{i=1}^{N_e}} \sum_{i=1}^{N_e} \Big[\underbrace{\|\boldsymbol{W}\boldsymbol{X}_i - \boldsymbol{W}_i\boldsymbol{X}_i\|_F^2}_{\text{Within-cluster reconstruction}} - \gamma \underbrace{\sum_{j\neq i} \|\boldsymbol{W}\boldsymbol{X}_i - \boldsymbol{W}_j\boldsymbol{X}_i\|_F^2}_{\text{Inter-expert diversity}} \Big], \gamma = \frac{1}{N_e - 1}$$

Step 2: Cluster-aware Expert Initialization

Practical solution: data-aware truncated SVD.

- Starting from copied dense weights $\forall i, \boldsymbol{W}_i \leftarrow \boldsymbol{W}$.
- Obtain the whitening matrix $\boldsymbol{S}_i$ via Cholesky decomposition: $\boldsymbol{S}_i \boldsymbol{S}_i^T = \boldsymbol{X}_i \boldsymbol{X}_i^T$
- Apply truncated SVD to $\boldsymbol{W}_i \boldsymbol{S}_i$.

$$\widetilde{\boldsymbol{W}}_i = T_{r_i}(\mathrm{SVD}(\underbrace{\boldsymbol{W}_i \boldsymbol{S}_i}_{\text{Data-aware transform}}))\boldsymbol{S}_i^{-1}$$

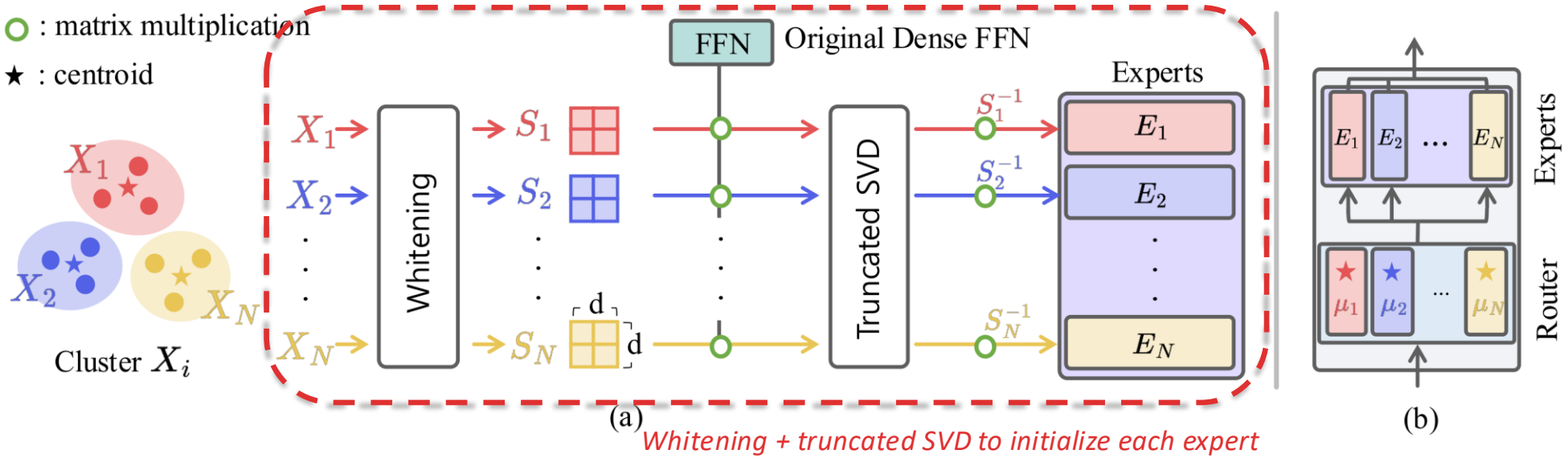
Truncated-SVD at effective rank r_i

Recover original weight space

- Truncation loss under the data distribution is given by the sum of squared discarded singular values.

$$\left\| \boldsymbol{W}_i \boldsymbol{X}_i - \widetilde{\boldsymbol{W}}_i \boldsymbol{X}_i \right\|_F^2 = \sum_{j > r_i} \sigma_{i,j}^2$$

Step 2: Cluster-aware Expert Initialization



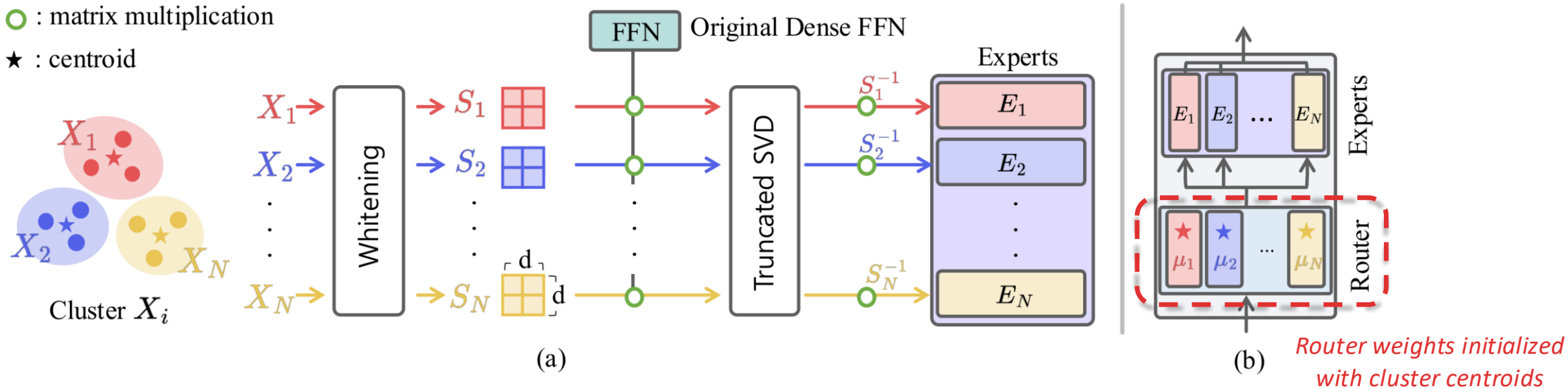
Whitening + truncated SVD to initialize each expert

Step 3: Cluster-aware Router Initialization

- Each expert corresponds to an activation cluster.
- Initialize router weights with the corresponding cluster centroids:

$$\boldsymbol{W}_r = [\boldsymbol{\mu}_1^T; \boldsymbol{\mu}_2^T; \ldots; \boldsymbol{\mu}_{N_e}^T]$$

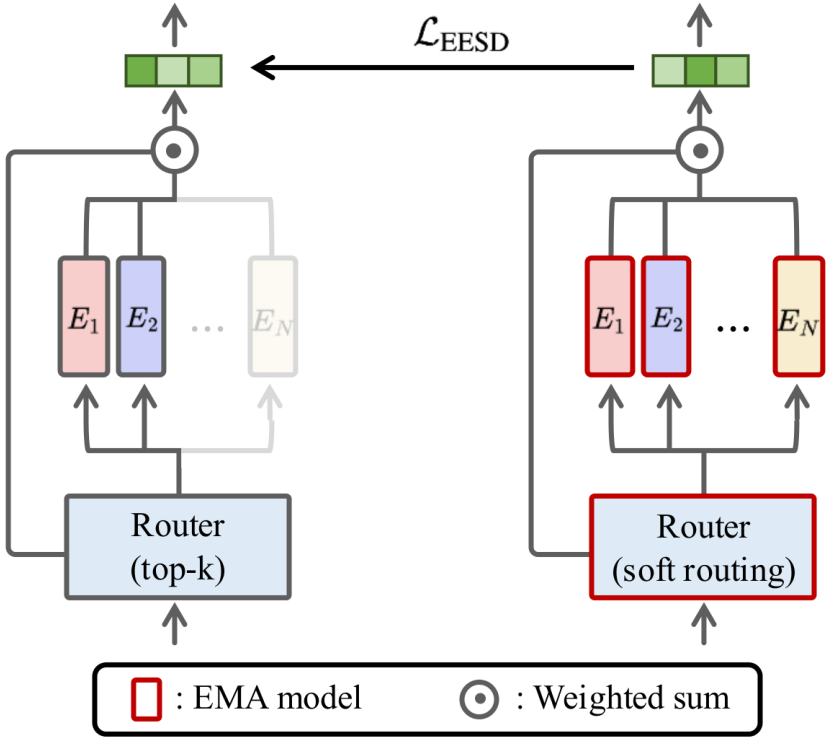
- Cluster centroids provide a meaningful routing prior for expert specialization from the onset of training.



Router weights initialized with cluster centroids

Expert-Ensemble Self-Distillation (EESD)

- Near-uniform routing probabilities imply unclear expert assignments.
- EESD uses a dense EMA ensemble as a stable target under high routing uncertainty.
- This provides stronger guidance for ambiguous tokens while preserving confident assignments.



$$y_{\text{MoE}}(\boldsymbol{x}) = \sum_{i \in \mathcal{T}_k(\boldsymbol{x})} \tilde{g}_i(\boldsymbol{x})\, E_i(\boldsymbol{x})$$

Student (sparse top-k MoE)

$$y_{\text{ens}}(\boldsymbol{x}) = \sum_{i=1}^{N_e} g_i^{\text{ema}}(\boldsymbol{x})\, E_i^{\text{ema}}(\boldsymbol{x})$$

Teacher (dense EMA ensemble)

$$\mathcal{L}_{\text{EESD}} = \frac{1}{T} \sum_{\boldsymbol{x}} \left\| \text{sg}(y_{\text{ens}}(\boldsymbol{x})) - y_{\text{MoE}}(\boldsymbol{x}) \right\|_2^2$$

Implementation Details

Architecture

- Dense backbone: CLIP ViT-B/32 and ViT-B/16.
- Every other FFN replaced by an MoE layer.
- 8 experts with top-2 token-choice routing.

Training

- Dense CLIP checkpoint: 4.0B seen samples on LAION-400M (15 epochs).
- MoE upcycling: additional 1.3B seen samples (5 epochs) .

Zero-shot Results

Recall@1 on MSCOCO retrieval & Accuracy on ImageNet variants & VTAB-Natural.

Model			MSCOCO			ImageNet-1k							VTAB
Arch.	MoE Init	Samples	I→T	T→I	Avg.	Val	V2	A	R	Sketch	ObjNet	Avg.	Natural
ViT-B/32													
Dense	-	4.0B	25.5	42.5	34.0	49.6	41.9	9.7	56.7	34.9	31.0	37.3	52.4
		4.0B+1.3B	30.8	47.5	39.2	56.7	48.5	13.9	64.0	41.2	36.1	43.4	58.3
MoE	Drop-Upcycling [30]	4.0B+1.3B	29.7	46.5	38.1	56.0	47.7	12.9	63.4	40.8	34.3	42.5	57.8
	Sparse Upcycling [20]		30.8	48.0	39.4	57.1	49.1	13.8	64.3	41.8	36.0	43.7	58.0
	CLIP-MoE [48]		29.5	46.8	38.2	56.6	48.1	**14.3**	64.2	41.4	35.7	43.4	58.8
	DeRS-LM [15]		**31.0**	47.7	39.4	56.8	48.6	13.9	64.2	41.1	36.4	43.5	58.1
	Cluster-aware Upcycling		**31.0**	**48.2**	**39.6**	**57.3**	**49.2**	14.0	**65.2**	**42.3**	**36.5**	**44.1**	**59.1**
ViT-B/16													
Dense	-	4.0B	32.5	49.1	40.8	59.4	51.7	20.1	67.3	42.9	39.4	46.8	58.1
		4.0B+1.3B	34.3	50.8	42.6	62.5	54.4	23.5	70.6	45.8	42.5	49.9	62.6
MoE	Drop-Upcycling [30]	4.0B+1.3B	34.1	51.3	42.7	62.0	54.5	22.7	70.8	45.7	42.9	49.8	60.9
	Sparse Upcycling [20]		34.9	50.9	42.9	63.0	**55.1**	23.7	71.2	46.3	42.3	50.3	62.0
	CLIP-MoE [48]		34.0	51.5	42.8	62.9	54.9	**24.5**	71.6	46.2	43.4	50.6	62.8
	Cluster-aware Upcycling		**35.4**	**51.6**	**43.5**	**63.2**	**55.1**	24.1	**72.1**	**46.8**	**43.5**	**50.8**	**63.3**

Few-shot and Full Fine-tuning

ImageNet-1k; 5-shot, 10-shot, full fine-tuning.

Model		ImageNet-1k		
Arch.	MoE Init	5-shot	10-shot	FT
Dense	-	50.4	57.1	72.8
MoE	Sparse Upcycling [20]	50.9	57.8	73.0
	Drop-Upcycling [30]	51.1	57.9	73.1
	CLIP-MoE [48]	51.3	58.0	73.2
	Cluster-aware Upcycling	**51.5**	**58.2**	**73.3**

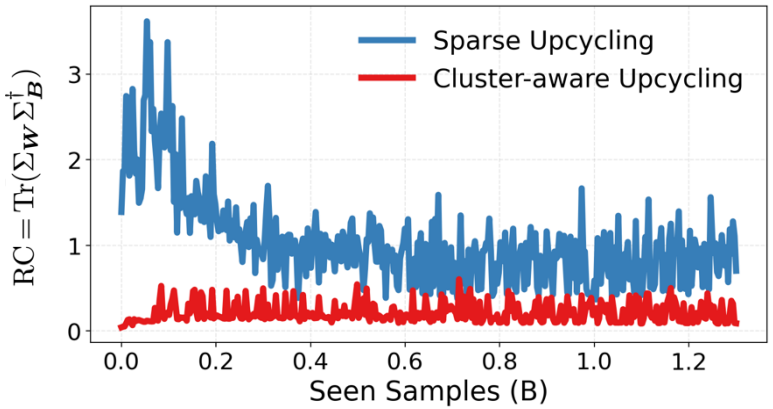
Ablation Study

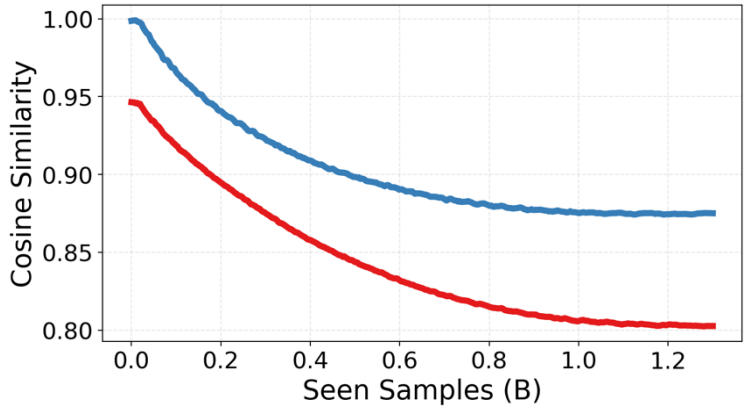
1. Cluster-aware initialization alone improves over sparse upcycling by breaking expert symmetry.
2. EESD provides stronger guidance under high routing uncertainty, but alone is insufficient.
3. Combining both works best: initialization induces specialization, and EESD preserves and reinforces it.

Model		MSCOCO		ImageNet-1k	
Cluster-init.	EESD	I→T	T→I	Val	10-shot
		34.9	50.9	63.0	57.8
✓		35.1	51.1	**63.2**	58.1
	✓	34.6	51.4	62.7	57.8
✓	✓	**35.4**	**51.6**	**63.2**	**58.2**

Analysis

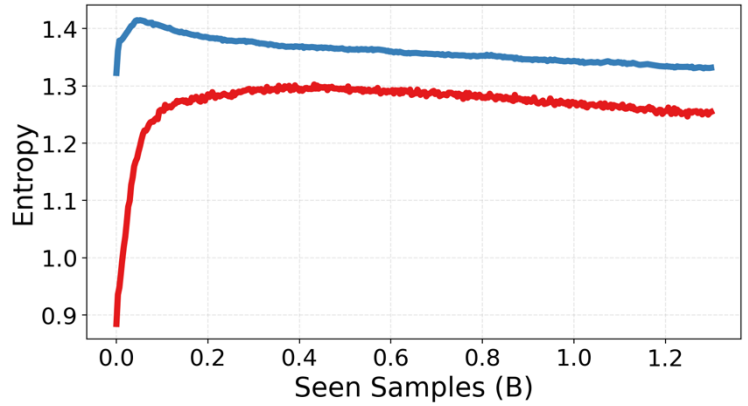
1. Relative Compactness ↓: more disentangled expert subspaces.

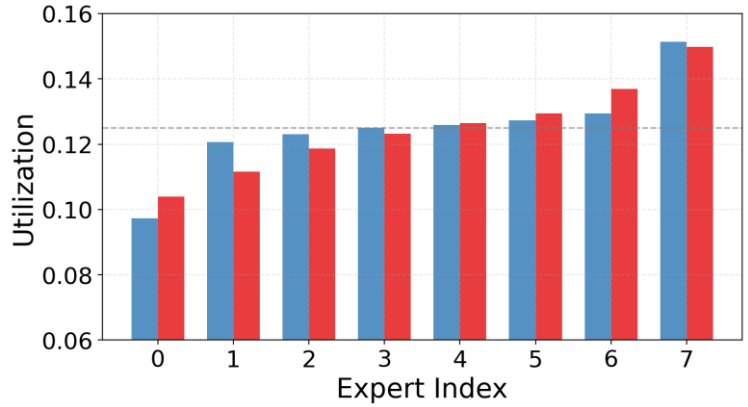




2. Expert Similarity ↓: higher parameter diversity across experts.

3. Routing Entropy: confident at initialization, then explores and restabilizes.





4. Expert Utilization: balanced expert allocation without routing collapse.

Conclusion

Cluster-aware Upcycling

- Beyond weight-copying; use pretrained activation geometry as an expert specialization prior.
- Dense activation clusters define expert roles, cluster centroids initialize the router, and EESD preserves the induced specialization.

Take-away: MoE upcycling benefits from using what the dense model already knows about the data!

Paper

Project page