Efficient Mixture of Experts (MoE) in LLMs

: From Routing to Compression

Sehyun Park

June 5, 2026

Seoul National University - IDEA Lab.

Outline

Dense MoE

Dense MoE

▶ Structure

- In the original MoE formulation (Jacobs et al., 1991; Jordan & Jacobs, 1994), all experts are evaluated for every input: a single network is replaced with N parallel experts and a gating network.
- Each expert is a standard FFN with independent parameters:

$$E_i(\boldsymbol{x}) = \mathrm{FFN}_i(\boldsymbol{x}), \qquad i = 1, \ldots, N \tag{1}$$

- The routing produces a softmax distribution over experts, and the output is a weighted sum of **all** N experts:

$$\boldsymbol{g}(\boldsymbol{x}) = \mathrm{softmax}(\boldsymbol{x}^\top \boldsymbol{W}_g) \in \mathbb{R}^N, \qquad \boldsymbol{y} = \sum_{i=1}^{N} g_i(\boldsymbol{x})\, E_i(\boldsymbol{x}) \tag{2}$$

※ $\boldsymbol{x} \in \mathbb{R}^d$: input representation; $\boldsymbol{W}_g \in \mathbb{R}^{d \times N}$: gating matrix.

▶ Objective

- Train end-to-end by adding a **balance loss** to the task loss to equalize expert load:

$$\mathcal{L} = \mathcal{L}_{\mathrm{task}} + \lambda\, \mathcal{L}_{\mathrm{bal}}, \qquad \mathcal{L}_{\mathrm{bal}} = \mathrm{CV}\big(\boldsymbol{I}\big)^2 = \frac{\mathrm{Var}_i(I_i)}{\big(\mathrm{Mean}_i(I_i)\big)^2} \tag{3}$$

※ $I_i = \sum_{\boldsymbol{x} \in \mathcal{B}} g_i(\boldsymbol{x})$: cumulative gating weight (importance) received by expert i over batch $\mathcal{B}$.

Sparse MoE

Sparse MoE: Token-based Routing

- **Motivation**
  - Dense MoE is computationally expensive at inference: all experts run for every token.
  - Token-based routing: the router decides, *per token*, which k experts to activate.

- **Top-k Gating** Shazeer et al. (2017); Switch Transformer (2022); Mixtral (2024)
  - Only top-k experts ($k \ll N$) are active per token. FLOPs reduce from $O(N \cdot T)$ to $O(k \cdot T)$.

$$G(\boldsymbol{x}) = \mathrm{softmax}(\mathrm{TopK}(\boldsymbol{x} \cdot \boldsymbol{W}_g + \boldsymbol{\varepsilon}))\,, \quad \boldsymbol{y} = \sum_{i=1}^{N} G_i(\boldsymbol{x}) \cdot E_i(\boldsymbol{x}) \tag{4}$$

  - Switch Transformer (Fedus et al., 2022): simplifies to $k = 1$, showing that single-expert routing is sufficient for large-scale pre-training.
  - Mixtral (Jiang et al., 2024): uses $k = 2$ with 8 experts; achieves performance comparable to larger dense models at a fraction of the active parameters.

Sparse MoE: Expert-based Routing

▶ Mixture-of-Experts with Expert Choice Routing

Zhou et al. (NeurIPS 2022)

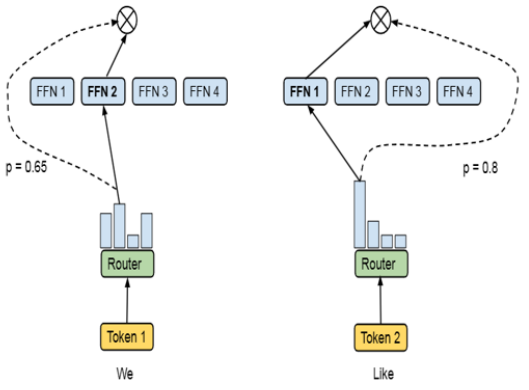
- Instead of tokens selecting experts, **each expert selects the top-k tokens** from the batch.

$$\boldsymbol{S} = \mathrm{Softmax}(\boldsymbol{X}\boldsymbol{W}_g) \in \mathbb{R}^{T\times N}, \boldsymbol{I}^{(i)} = \mathrm{TopK}(\boldsymbol{S}_{:,i})$$

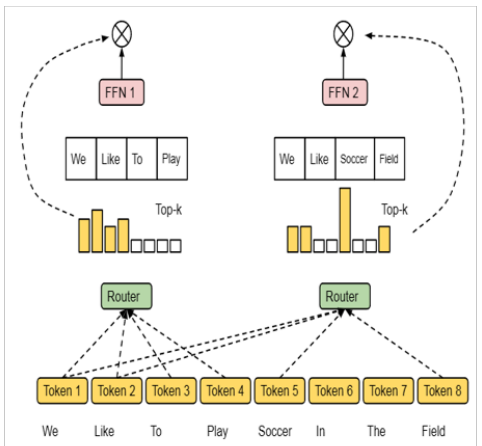
※ $\boldsymbol{X} \in \mathbb{R}^{T\times d}$: token matrix, $\boldsymbol{I}^{(i)}$: index set of tokens selected by expert i.

▶ Experiment Results

- Empirically achieves $> 2\times$ faster training convergence compared to Switch Transformer (top-1) and GShard (top-2) under the same compute budget.



〈Token-based MoE〉



〈Expert-based MoE〉

Sparse-to-Dense

Sparse-to-Dense: Motivation

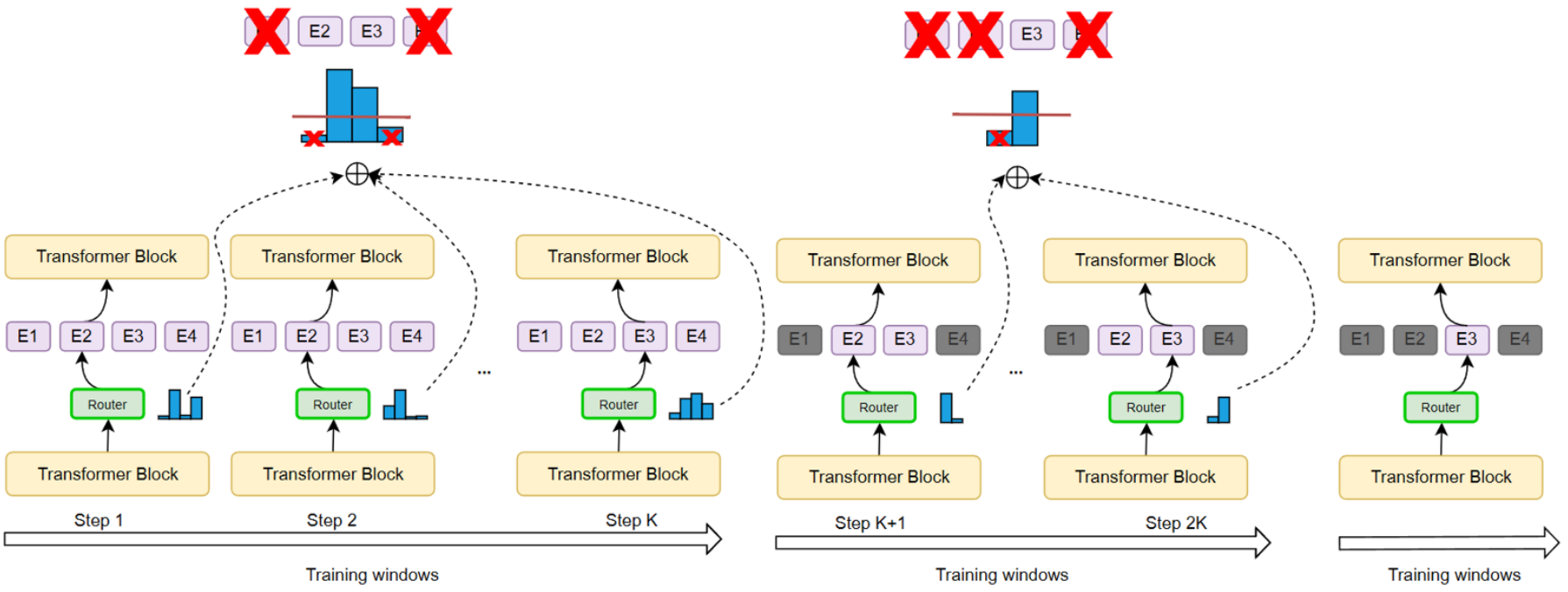
- Sparse MoE reduces *compute* per token, but all expert parameters must reside in memory.
- A Mixtral-8×7B model stores ≈46.7B parameters; deploying it requires ∼87GB GPU memory.
- ▶ Sparse-to-Dense research compresses a trained Sparse MoE into one or a small number of dense models:

Method	Key idea
Pruning	Remove low-importance experts
Knowledge Distillation	Train a single FFN student
Expert Merging (Sub-MoE)	Merge experts

Sparse-to-Dense: Pruning

▶ TSEP: Task-Specific Expert Pruning

Chen et al. (2022)



〈The pipeline of Task-Specific Exeprt Pruning〉

Sparse-to-Dense: Pruning

▶ TSEP: Task-Specific Expert Pruning

Chen et al. (arxiv, 2022); Sarkar et al. (NeurIPS, 2024)

- Observation: for a specific downstream task, only a subset of experts are consistently activated; the rest contribute little.
- An *expert proficiency score* is defined per expert i on task dataset $\mathcal{D}$:

$$\text{score}(E_i) = \sum_{\boldsymbol{x} \in \mathcal{D}} G_i(\boldsymbol{x}) \tag{5}$$

※ $G_i(\boldsymbol{x})$ is indicator that expert i is selected for input $\boldsymbol{x}$.

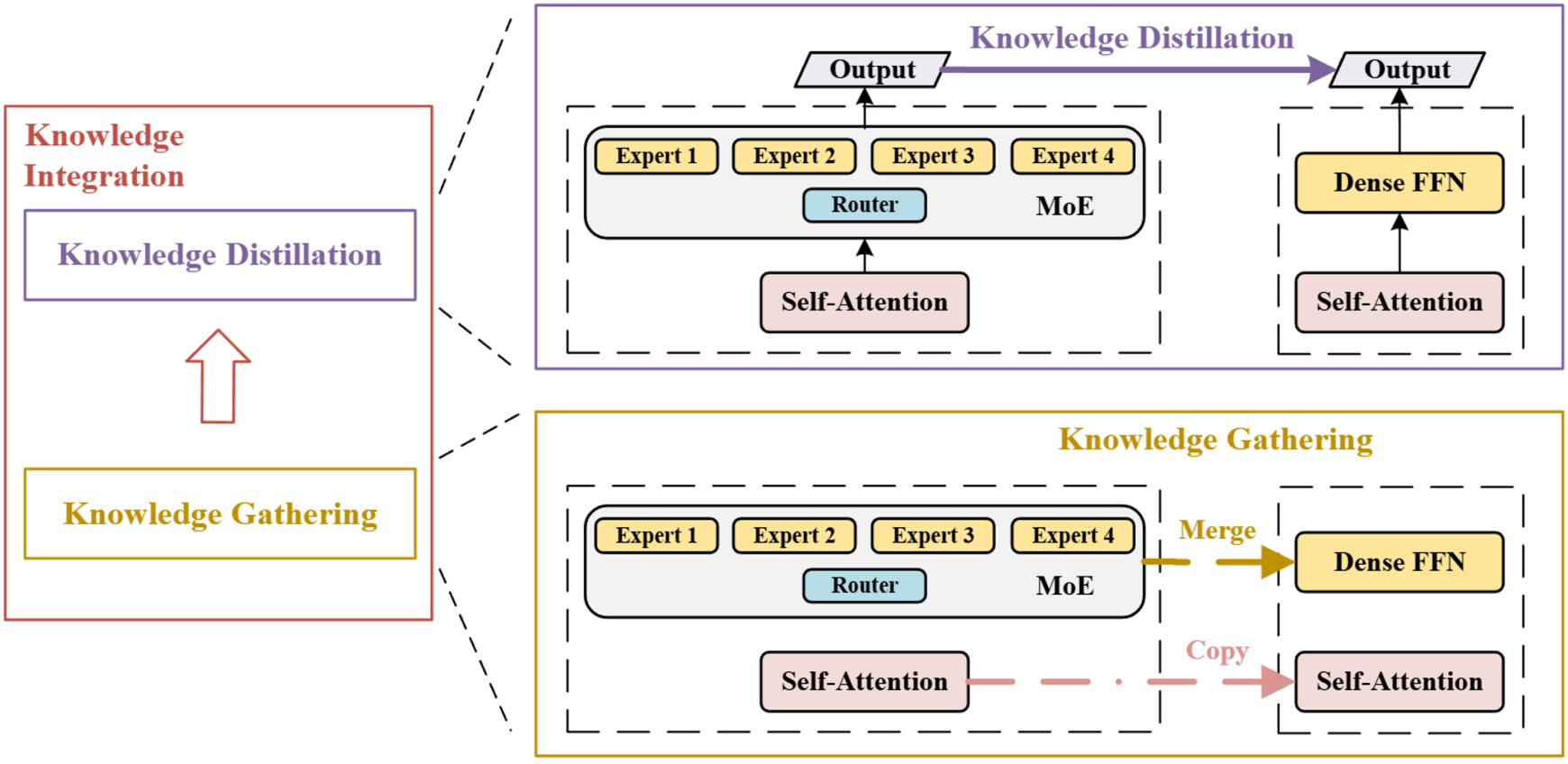
▶ Results

- The resulting single-expert model preserves 99.3% of MoE performance across six downstream tasks while achieving $2\times$ inference speedup by eliminating expert parallelism overhead.
- However, Sarkar et al. (2024) show that naive activation-based pruning fails to recover the performance of a scratch-trained SMoE of equivalent size.

Sparse-to-Dense: Knowledge Distillation

▶ OneS: One Student Knows All Experts Know

Xue et al. (arxiv, 2022)



〈An overview of OneS〉

Sparse-to-Dense: Knowledge Distillation

- **OneS: One Student Knows All Experts Know**

  Xue et al. (arxiv, 2022)

  - Goal: obtain a **single dense student model** that matches the knowledge of a sparse MoE teacher.
  - Two-stage framework:
    1. **Knowledge Gathering**: aggregate N expert weights $\{\boldsymbol{W}_i\}_{i=1}^{N}$ into one set of parameters:
       - SVD-KG: $\boldsymbol{W}_{\text{student}} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{U}_{r,i} \boldsymbol{\Sigma}_{r,i} \boldsymbol{V}_{r,i}^{\top}$,

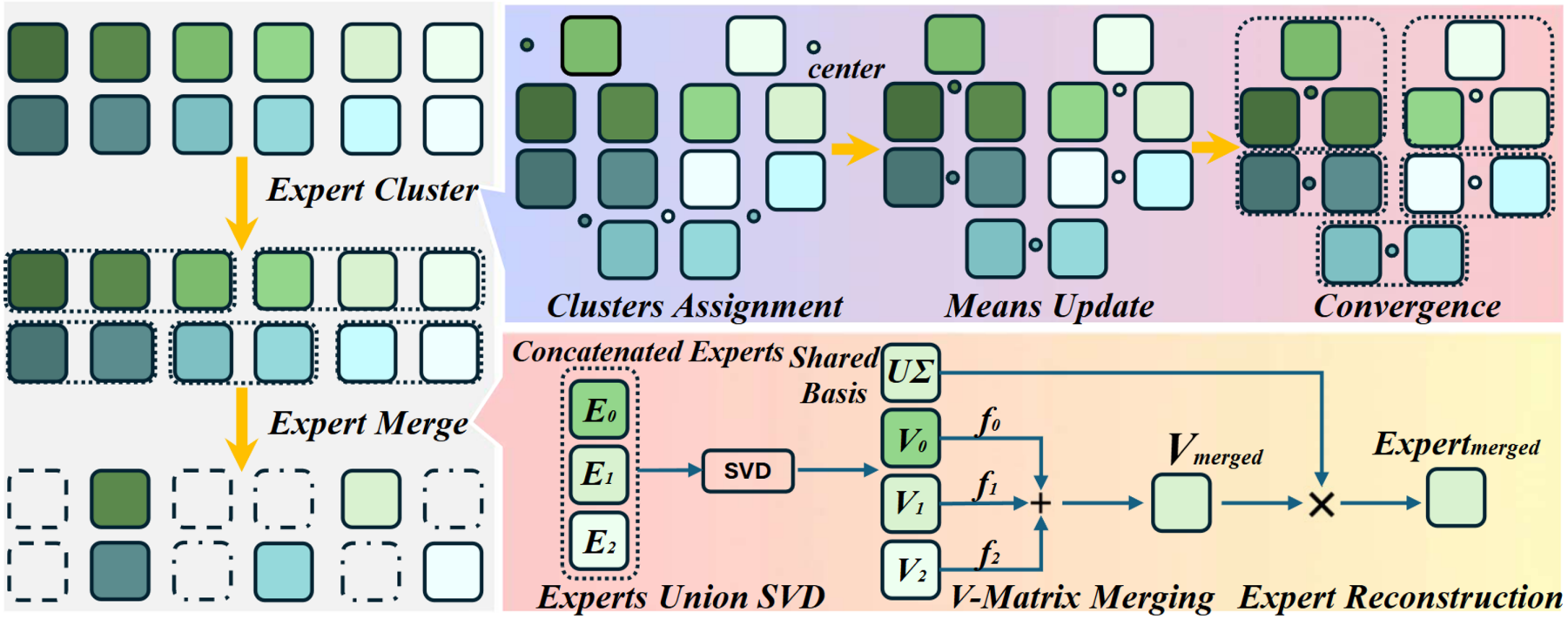
         where $\boldsymbol{W}_i \approx \boldsymbol{U}_{r,i} \boldsymbol{\Sigma}_{r,i} \boldsymbol{V}_{r,i}^{\top}$ is the rank-r SVD approximation of the i-th expert
    2. **Knowledge Distillation**: train the student to minimize the following objective:

$$\mathcal{L} = (1 - \lambda)\, \mathcal{L}_{\text{CE}} + \lambda\, \text{KL}(p_{\text{teacher}}(\boldsymbol{x}; T) \,\|\, p_{\text{student}}(\boldsymbol{x}; T)) \tag{6}$$

Sparse-to-Dense: Expert Merging (Sub-MoE)

▶ Sub-MoE: Efficient Mixture-of-Expert LLMs Compression

Li et al. (AAAI 2026)



〈An overview of Sub-MoE〉

Sparse-to-Dense: Expert Merging (Sub-MoE)

▶ **Sub-MoE: Efficient Mixture-of-Expert LLMs Compression**

Li et al. (AAAI 2026)

- Rather than producing a single dense model, Sub-MoE merges groups of experts into fewer experts within the MoE structure, preserving the routing mechanism.
- Core problem: direct weight averaging causes *parameter conflicts* because experts operate in distinct representation spaces (inter-expert similarity ≈ 0.1–0.3 in Mixtral-8×7B).

▶ **Stage 1: Adaptive Expert Clustering**

- Experts that respond similarly to the same inputs are better merge candidates. Functional similarity is measured as:

$$\mathrm{Sim}(E_i, E_j) = \frac{1}{|\mathcal{X}|} \sum_{\boldsymbol{X} \in \mathcal{X}} \frac{1}{|\boldsymbol{X}|} \sum_{\boldsymbol{x} \in \boldsymbol{X}} \frac{E_i(\boldsymbol{x}) \cdot E_j(\boldsymbol{x})}{\|E_i(\boldsymbol{x})\| \, \|E_j(\boldsymbol{x})\|} \tag{7}$$

- Experts are grouped via **K-means clustering** on this similarity.

Sparse-to-Dense: Expert Merging (Sub-MoE)

- Stage 2: Subspace Expert Merging

  - **Experts Union Decomposition**: for each cluster, concatenate expert weights and apply joint SVD to find a shared subspace:

$$\mathrm{SVD}([\boldsymbol{W}_1;\cdots;\boldsymbol{W}_n]) = \boldsymbol{U}\boldsymbol{\Sigma}[\boldsymbol{V}^{(1)};\cdots;\boldsymbol{V}^{(n)}]^\top \tag{8}$$

    The shared $\boldsymbol{U}$ aligns all experts; conflicts are isolated to the expert-specific $\boldsymbol{V}^{(i)}$.
  - **Frequency-based V-Matrix Merging**: merge expert-specific components weighted by router activation frequency:

$$f_i(\boldsymbol{X}) = \frac{\sum_{\boldsymbol{x}\in\boldsymbol{X}} \mathbb{I}[i \in \mathrm{TopK}(G(\boldsymbol{x}))]}{|\boldsymbol{X}|}, \qquad \boldsymbol{V}_{\mathrm{merged}} = \frac{\sum_{i=1}^{N} f_i(\boldsymbol{X}) \cdot \boldsymbol{V}^{(i)}}{\sum_{i=1}^{N} f_i(\boldsymbol{X})} \tag{9}$$

  - **Expert Reconstruction**:

$$\boldsymbol{W}_{\mathrm{merged}} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}_{\mathrm{merged}}^\top \tag{10}$$

  - On Mixtral-8×7B: 50% expert reduction with 86% performance retained.

End