

Improving Parameter Utilization by Sharing Neural Experts Across Layers in Transformers

Dian Jiao*, Jiaxin Duan*, Shuai Zhao

China Electronics Cloud Technology Co., Ltd.

554848018@qq.com, jiaxdan@gmail.com, t.zhaosh@gmail.com

<https://github.com/CESTC-REAL/Self-MoE>

Abstract

Transformer-based large language models often suffer from inter-layer parameter redundancy, where functional transformations are redundantly learned across network depths. We propose CS-MoE, a novel Transformer architecture featuring cross-layer expert sharing to address this inefficiency. Deviating from the widely used Mixture-of-Experts (MoE) architecture that terminates each Transformer block with layer-isolated experts, CS-MoE combines layer-independent experts with concurrent access to a centralized, globally shared expert pool. This *Global Experts Sharing* mechanism enables elastic control over token-level parameter activation and computational consumption (FLOPs). Experiments demonstrate that CS-MoE achieves lower perplexity than equal-scale dense Transformers while activating only 55% of parameters. Furthermore, its performance scales monotonically with an increased number of activated experts and approaches MoE counterparts that consume more FLOPs by expanding the shared pool without increasing FLOPs. CS-MoE also establishes a flexible Pareto frontier between computational cost and model capacity, offering an efficient alternative for computation-constrained environments.

1 Introduction

The proliferation of Transformer-based architectures has catalyzed the success of Large Language Models (LLMs), but scaling toward trillion-parameter regimes yields diminishing marginal returns in efficiency (Dai et al., 2024; Muennighoff et al., 2025). Empirical inquiries into weight pruning, model compression (Shazeer et al., 2017; Lepikhin et al., 2020), and activation dynamics suggest that dense networks suffer from significant activation redundancy, wherein a substantial portion of Feed-Forward Network (FFN) parameters

*Equal contributions.

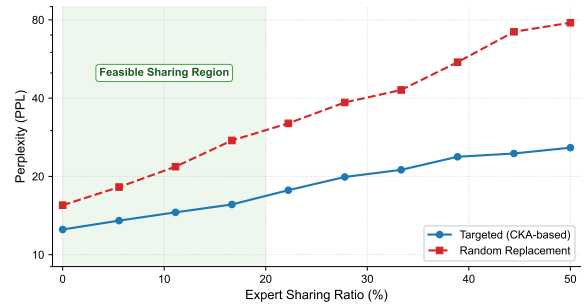


Figure 1: PPL of MoE models with random expert replacement and deliberate expert replacement based on CKA analysis.

contributes negligibly to the final output for any given input token (Contributors, 2025). This latent underutilization leads to a suboptimal ratio of information density to computational expenditure during inference.

The Mixture-of-Experts (MoE) paradigm has emerged as a standard solution, substituting dense layers with sparse experts and selectively activating a subset (Top- K) per token to decouple total model capacity from active computation (FLOPs) (Yun et al., 2024; Jiang et al., 2024). However, conventional MoE designs rely on layer-isolated experts, introducing parameter redundancy by preventing the network from exploiting functional recurrence across different depths (Jin et al., 2024; Wen et al., 2025; Yang et al., 2024). To investigate this, we conduct a pilot study replacing deep-layer experts with functionally equivalent shallow counterparts identified via Centered Kernel Alignment (CKA) (details in Appendix B). As illustrated in Figure 1, this CKA-based expert substitution incurs only moderate perplexity degradation and substantially outperforms random replacement, indicating that experts are largely interchangeable across depths.

Based on these insights, we propose CS-MoE, a novel Transformer architecture that bypasses layer isolation via a *Global Experts Sharing* mechanism.

First, a *Fixed Path* comprising a small set of *Independent Experts* is unconditionally activated at each layer to capture depth-specific features such as hierarchical syntax. Additionally, a *Dynamic Path* grants all layers concurrent access to a global expert pool. By employing a layer-specific router to select proportional experts from this global repository, CS-MoE preserves the hierarchical integrity of the Transformer while maximizing parameter reutilization.

In experiments, we validate the efficacy of CS-MoE across model scales ranging from 0.6B to 12B parameters. Extensive results demonstrate its three-fold advantages: 1) CS-MoE achieves superior parameter efficiency, outperforming dense baselines in perplexity while requiring only 55% active parameters. 2) Its performance scales monotonically with the expert activation count (K), leveraging the capacity of the shared pool; 3) The architecture offers a flexible Pareto frontier, providing a highly efficient alternative in FLOP-constrained environments equivalent to the performance of standard MoE by scaling the shared experts pool.

2 Methodology

2.1 Formalization and Topology

We formalize the Transformer Feed-Forward Network through a generalized Mixture-of-Experts (MoE) framework. In this view, a standard *dense Transformer* represents a constrained instance of this topology with a single, active expert ($N = 1, K = 1$) per layer. A *sparse MoE Transformer* scales parameter capacity via a larger pool of experts ($N > 1$) while restricting active computation to a small subset ($K \ll N$). *CS-MoE* generalizes these paradigms into a dual-tier topology. It anchors a localized set of experts to specific layers to capture depth-dependent features, while providing a centralized, shared expert pool accessible across all layers. This design allows individual experts to participate in multiple stages of the network, bypassing traditional layer isolation to maximize parameter utility and computational efficiency.

2.2 CS-MoE Architecture

As illustrated in Figure 2, CS-MoE restructures each Transformer layer $l \in \{1, \dots, L\}$ into a dual-path execution model: a **Fixed Path** consisting of layer-specific Independent Experts and a **Dynamic Path** drawing from a global Shared Expert Pool. The total expert set available to the network is par-

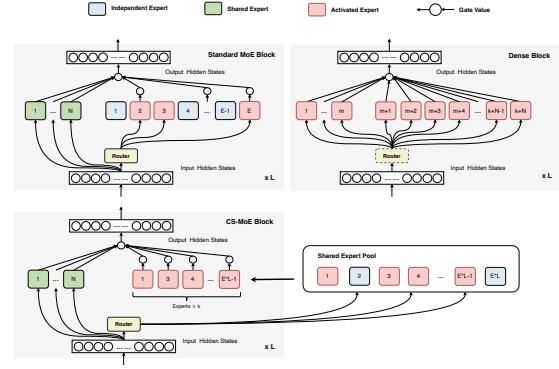


Figure 2: Overview of Transformer architectures: MoE (upper left), dense (upper right), and our CS-MoE (lower). The dashed box indicates a virtual router.

itioned as follows:

$$\mathcal{E}_{total} = \left(\bigcup_{l=1}^L \mathcal{E}_{indep}^{(l)} \right) \cup \mathcal{E}_{shared}$$

where $\mathcal{E}_{indep}^{(l)} = \{e_1^{(l)}, \dots, e_N^{(l)}\}$ contains N independent experts private to layer l , and $\mathcal{E}_{shared} = \{s_1, \dots, s_M\}$ is a large, centralized repository of M experts accessible by every layer in the Transformer stack. In this configuration, the execution logic for each layer follows two distinct principles:

- **Routing-free Fixed Path:** Each layer l activates its own N Independent Experts ($\mathcal{E}_{indep}^{(l)}$) unconditionally for every token. This ensures a baseline of depth-specific transformation capacity without the overhead of routing, effectively maintaining the reliability of dense architectures.
- **Sparse Dynamic Path:** The routing mechanism in CS-MoE is exclusively restricted to the *Shared Expert Pool*. For each token at layer l , a layer-specific router selects and activates k experts from $\mathcal{E}_{shared}$. Crucially, because the shared pool is universal, a specific shared expert s_j may be activated simultaneously across multiple layers for the same or different tokens, allowing for extreme longitudinal parameter reuse.

Consequently, the total computation of a single layer is composed of N independent experts and k routed shared experts, resulting in a total of $k + N$ activated experts per layer. This hybrid structure allows CS-MoE to simulate the capacity of an ultra-wide MoE while maintaining a lean physical parameter footprint.

2.3 Hybrid Activation and Routing Strategy

The execution of a CS-MoE layer is defined by the simultaneous processing of these two paths. Given a token representation $\mathbf{h}_l \in \mathbb{R}^d$, the routing mechanism manages only the dynamic path (the shared pool).

Shared Path Routing: The layer-specific router R_l computes scores s_l for the M shared experts:

$$\mathbf{s}_l = W_{r,l} \mathbf{h}_l$$

where $W_{r,l} \in \mathbb{R}^{M \times d}$. We then apply a Top- K gating function to select the most relevant shared experts:

$$\mathbf{g}_l = \text{Softmax}(\text{TopK}(\mathbf{s}_l, K))$$

Output Aggregation: The final output of the layer, $\mathbf{o}_l$, is the sum of the dense independent experts and the sparsely activated shared experts:

$$\mathbf{o}_l = \underbrace{\sum_{j=1}^{N_{\text{indep}}} \text{Expert}_{e,j}^{(l)}(\mathbf{h}_l)}_{\text{Fixed Path}} + \underbrace{\sum_{i \in \text{TopK}} g_{l,i} \cdot \text{Expert}_{s,i}(\mathbf{h}_l)}_{\text{Dynamic Path}}$$

This design ensures that every token benefits from N_{indep} specialized parameters tailored to the current layer’s depth, while simultaneously drawing from K experts in the global pool. By adjusting K and M , the architecture can precisely tune the ratio of compute-to-parameters, effectively breaking the linear relationship between model depth and parameter count.

2.4 Layer-wise Load Balancing

To prevent the shared pool from suffering from expert collapse - where a few experts dominate the routing across all layers - we implement a layer-wise auxiliary load-balancing loss $\mathcal{L}_{aux}^{(l)}$. This loss is calculated specifically for the shared expert pool at each layer to ensure diverse utilization.

For a batch of T tokens, the balancing loss is defined as:

$$\mathcal{L}_{aux}^{(l)} = M \sum_{i=1}^M f_{l,i} \cdot P_{l,i}$$

Where $f_{l,i}$ is the fraction of tokens in the batch that selected shared expert i at layer l , and $P_{l,i}$ is the average routing probability assigned to that expert. The total objective function is:

$$\mathcal{L}_{total} = \mathcal{L}_{LM} + \alpha \sum_{l=1}^L \mathcal{L}_{aux}^{(l)}$$

where α is the balancing coefficient. This ensures that the shared pool remains a robust, multi-functional resource throughout the training process.

3 Experiments

3.1 Experimental Setup

We evaluate the efficiency, scalability, and structural convergence of **CS-MoE** across scales ranging from 0.6B to 12B parameters using the Qwen3 architecture (Yang et al., 2025) as our experimental backbone. Detailed architectural dimensions and layer allocations are provided in Appendix C.

Efficiency Metric To quantify parameter reuse within our dual-path design, we define the **Expert Utilization Ratio (EUR)** (ρ):

$$\rho = \frac{|\bigcup_{l=1}^L S_l|}{\min(M, \delta)},$$

where M is the physical capacity of the shared pool, S_l is the set of shared experts activated at layer l , and $\delta = \sum_{l=1}^L |S_l|$ is the cumulative expert query volume. The term $\min(M, \delta)$ ensures normalization across different allocation budgets.

3.2 Architectural Efficiency and Scalability

Our pre-training workflow utilizes an integrated high-quality corpus of **WuDao** (Yuan et al., 2021) and **DCLM** (Li et al., 2024b) for 12,500 steps with an effective batch size $\approx 1\text{M}$ tokens. Perplexity (PPL) is used to trace training dynamics.

Parameter Expansion vs. Dense Baselines We compare CS-MoE directly against Dense Transformers under matched parameter and FLOP budgets. As illustrated in Figure 3, CS-MoE consistently achieves lower pre-training perplexity across all model scales (exact numerical values in Appendix D.1, Table 7). Crucially, CS-MoE exhibits a strong “Parameter Expansion” effect: an 8B CS-MoE model activating only **55%** of its physical parameters per forward pass outperforms a fully activated 8B Dense baseline (PPL 9.55 vs. 9.75), demonstrating that cross-layer expert reuse expands effective representational capacity without increasing active FLOPs.

Dynamic Compute Allocation In Figure 4, we evaluate the scaling behavior of CS-MoE by increasing the Top- K activation count while keeping the parameter count fixed. The results demonstrate monotonic perplexity reductions as active compute

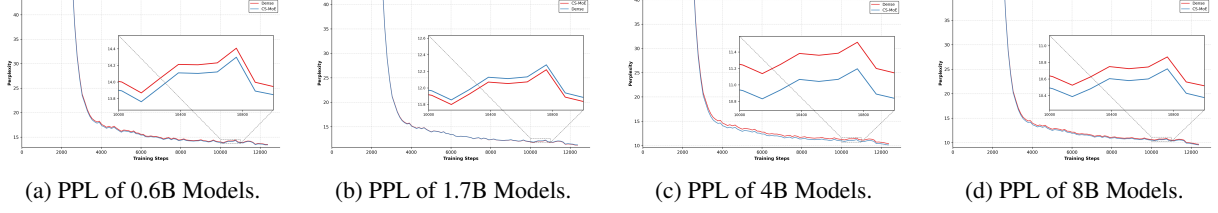


Figure 3: Comparison between CS-MoE and Dense models.

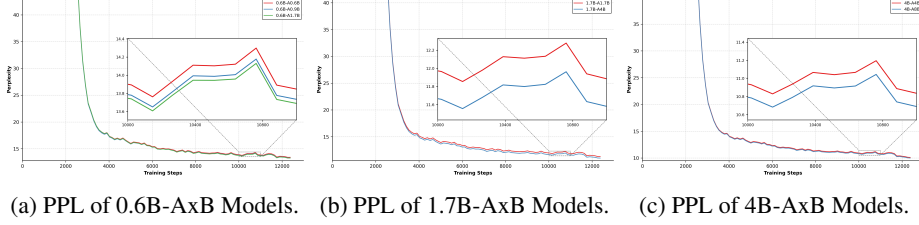


Figure 4: Comparison among CS-MoE models with varying activations.

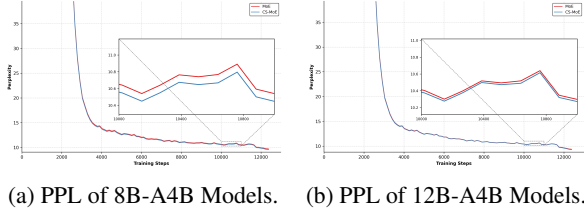


Figure 5: Comparison between CS-MoE and Sparse MoE models with equal activations.

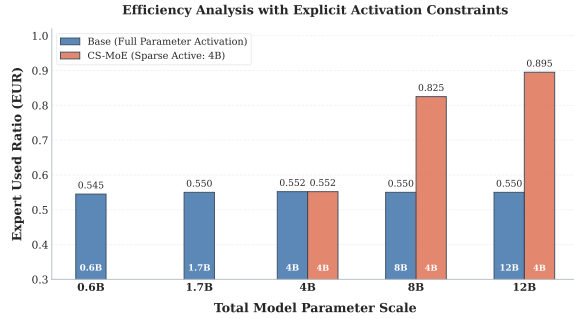


Figure 6: Efficiency analysis of CS-MoE models.

increases (e.g., 0.6B-A1.7B reaches 13.11 PPL vs. 13.48 for 0.6B-A0.6B). Unlike rigid dense models, CS-MoE allows dynamic FLOP scaling at inference time by tuning the routing budget into the centralized pool.

Convergence to Sparse MoE Baselines In Figure 5, we strictly align both the total parameter footprint (HBM storage) and active FLOPs between CS-MoE and standard MoEs with layer-isolated experts. CS-MoE achieves lower perplexity (9.65 vs. 9.70 at 8B scale, 9.19 vs. 9.40 at 12B scale). As depicted in Figure 6, expanding the shared pool

Table 1: **Downstream benchmark accuracy (%)** on C-Eval and AQuA across training checkpoints.

Steps	0.6B / -A0.9B		1.7B / -A4B		4B / -A8B	
	Dense	CS-MoE	Dense	CS-MoE	Dense	CS-MoE
C-Eval (Multi-discipline Chinese Evaluation)						
2,500	28.2	31.5	30.1	31.7	29.3	29.8
7,500	35.1	39.3	36.4	40.1	35.6	41.0
12,500	37.5	41.9	38.6	42.3	40.7	43.2
AQuA (Multi-step Algebraic Reasoning)						
2,500	22.4	26.1	21.8	28.5	27.2	33.6
7,500	29.1	34.0	29.8	34.3	36.1	37.2
12,500	31.2	36.4	32.6	37.8	39.1	40.5

size (M) drives the EUR (ρ) toward 1.0, asymptotically recovering the capacity upper bound of standard MoEs while maintaining flexible parameter sharing.

3.3 Evaluation on Downstream Benchmarks

We evaluate cross-scale downstream generalization on **C-Eval** (Nguyen et al., 2024) (a comprehensive Chinese benchmark spanning 52 domains) and **AQuA** (Ling et al., 2017) (multi-step algebraic reasoning). As summarized in Table 1, CS-MoE consistently surpasses the Dense baseline across all evaluated parameter scales and intermediate checkpoints. Crucially, the performance advantage is most pronounced under severe parameter constraints at the 0.6B scale, delivering final gains of +4.4% on C-Eval and +5.2% on AQuA; this highlights that concurrent access to a centralized shared pool effectively compensates for the restricted capacity of smaller models. Furthermore, CS-MoE exhibits substantially accelerated conver-

Table 2: Downstream performance comparison at 8B scale after full-scale pre-training on 4.5 trillion tokens.

Benchmark	8B Dense Baseline	8B-A8B CS-MoE (Ours)	Diff (Δ)
BBH (Suzgun et al., 2023)	0.2867	0.7507	+0.4640
CMMLU (Li et al., 2024a)	0.7417	0.7942	+0.0525
ARC (Clark et al., 2018)	0.6875	0.7300	+0.0425
GSM8K (Cobbe et al., 2021)	0.9400	0.9450	+0.0050
C-Eval (Huang et al., 2023)	0.6996	0.7038	+0.0042
IFEval (Zhou et al., 2023)	0.4300	0.4005	-0.0295
MATH (Hendrycks et al., 2021)	0.7930	0.6660	-0.1270

gence dynamics, matching or exceeding the Dense baseline’s final performance (12,500 steps) within merely 5,000 to 7,500 training steps, confirming that cross-layer expert sharing expedites knowledge acquisition throughout the pre-training lifecycle.

To evaluate long-term convergence and scaling stability, our 8B CS-MoE model is subjected to full-scale pre-training across a 4.5-trillion-token corpus expanded with **Ultra FineWeb** (Wang et al., 2025) (details in Appendix C). The models were evaluated zero/few-shot across seven diverse benchmarks spanning multi-step reasoning (BBH (Suzgun et al., 2023), GSM8K (Cobbe et al., 2021)), domain knowledge (CMMLU (Li et al., 2024a), C-Eval (Huang et al., 2023), ARC (Clark et al., 2018)), instruction following (IFEval (Zhou et al., 2023)), and complex mathematics (Competition Math (Hendrycks et al., 2021)).

As shown in Table 2, CS-MoE achieves substantial improvements over the Dense baseline on complex reasoning (BBH +0.4640), world knowledge (CMMLU +0.0525, ARC +0.0425), and mathematical problem-solving (GSM8K +0.0050). We observed steady convergence with zero expert collapse throughout the 4.5T-token trajectory. Minor trade-offs observed on Competition Math and IFEval can be attributed to raw pre-training alignment sensitivity, which is typically addressed during targeted supervised fine-tuning.

3.4 Ablation Studies

To isolate the contributions of individual architectural choices, we conducted systematic ablation experiments under strictly controlled parameter and activation budgets at the 4B scale, as summarized in Table 3. First, evaluating the independent expert allocation reveals that anchoring at least one layer-private expert ($N_{indep} = 1$) is essential for capturing depth-specific syntactic representations; eliminating it entirely ($N_{indep} = 0$) causes severe performance degradation (10.32 vs. 10.08 PPL), while allocating additional private experts ($N_{indep} = 2$) yields diminishing returns due to

Table 3: Ablation results of architectural components under matched 4B parameters and 4B activations.

Ablation Setting	PPL $\downarrow$	C-Eval $\uparrow$	AQuA $\uparrow$
1. Independent Expert Allocation (N_{indep})			
$N_{indep} = 0$ (Shared pool only)	10.32	41.8	39.5
$N_{indep} = 1$ (Default)	10.08	43.2	40.5
$N_{indep} = 2$	10.11	43.0	40.3
2. Router Architecture			
Centrally Shared Router	10.29	41.9	39.6
Layer-Specific Router (Default)	10.08	43.2	40.5
3. Load Balancing Coefficient (α)			
$\alpha = 0.0$ (No balancing / Collapse)	10.35	41.4	39.2
$\alpha = 0.01$ (Default)	10.08	43.2	40.5
$\alpha = 0.1$	10.13	42.8	40.1
$\alpha = 0.5$	10.12	42.0	39.6
4. Cross-Layer Parameter Tying Paradigm			
Static Parameter Tying (ALBERT-style)	12.74	38.4	37.6
Standard Dense (Untied Baseline)	10.45	40.7	39.1
CS-MoE Dynamic Sharing (Ours)	10.08	43.2	40.5

shrinking the shared pool capacity. Second, our layer-specific routing policy markedly outperforms a globally shared router (10.08 vs. 10.29 PPL), demonstrating that different network depths require customized gating decisions to process hierarchical abstractions. Third, tuning the auxiliary load-balancing coefficient demonstrates that while omitting balancing loss ($\alpha = 0$) triggers expert collapse (10.35 PPL), an intermediate penalty ($\alpha = 0.01$) effectively balances pool utilization without over-constraining the model’s natural routing preferences. Finally, benchmarking against classical cross-layer parameter tying illustrates the fundamental advantage of our design: while rigid ALBERT-style weight sharing severely impairs representational capacity (12.74 PPL), CS-MoE’s dynamic routing reuses weights adaptively across layers, surpassing even the standard untied Dense baseline (10.45 PPL) while maintaining high parameter efficiency.

4 Conclusion

We presented **CS-MoE**, a Transformer architecture featuring cross-layer expert sharing. By coupling depth-localized independent experts with a globally shared expert pool, CS-MoE breaks inter-layer parameter isolation while preserving hierarchical representation. Extensive evaluations across model scales and training regimes demonstrate its lower perplexity, higher parameter utilization, and superior downstream task performance compared to dense and sparse baselines.

Limitations

While CS-MoE demonstrates superior parameter efficiency and promising scaling properties, several limitations and open challenges remain to be addressed in future work:

- **Generalization at Ultra-Large Model Scales:** Due to constrained computational and hardware resources, our empirical validation is conducted up to the 12B scale. The scaling dynamics, routing stability, and Pareto efficiency of CS-MoE on ultra-large regimes (e.g., 70B+ parameters) remain to be rigorously verified.
- **Distributed Infrastructure and Acceleration:** The architectural topology of CS-MoE fundamentally departs from both dense Transformers and standard layer-isolated MoEs. Current mainstream distributed frameworks (e.g., standard *Expert Parallelism* and *Pipeline Parallelism*) are not tailored for a centralized, cross-layer shared expert pool. While CS-MoE possesses significant untapped potential for training throughput and inference acceleration via cross-layer weight reuse and caching, fully unlocking these speedups requires dedicated system-level infrastructure, such as custom communication primitives and optimized routing kernels.
- **Training Paradigms and Adaptation Techniques:** This study strictly evaluates CS-MoE trained under the supervision of next-token prediction. The behavior and efficacy of cross-layer expert sharing under post-training alignment pipelines (e.g., RLHF, DPO, or test-time reinforcement learning) have not yet been established. Additionally, downstream parameter-efficient fine-tuning (PEFT) strategies (e.g., LoRA) warrant specialized study, as naively adapting or tying shared experts across different network depths may require depth-aware adaptation mechanisms.

References

- Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. PaLM: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. [Think you have solved question answering? try ARC, the AI2 reasoning challenge.](#) *CoRR*, abs/1803.05457.
- 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. 2021. [Training verifiers to solve math word problems.](#) *CoRR*, abs/2110.14168.
- OpenReview Contributors. 2025. Sparseswaps: Tractable llm pruning mask refinement at scale. In *The Thirteenth International Conference on Learning Representations*.
- Damai Dai, Chengqi Deng, Shuo Zhao, et al. 2024. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*.
- Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. 2019. Universal transformers. In *International Conference on Learning Representations*.
- William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39.
- Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced BERT with disentangled attention. In *International Conference on Learning Representations*.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. [Measuring mathematical problem solving with the MATH dataset.](#) In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.
- Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, Yao Fu, Maosong Sun, and Junxian He. 2023. [C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models.](#) In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.
- Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, L  lio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Th  ophile Gerv  t, Thibaut Lavril, Thomas Wang, Timoth  e Lacroix, and William El Sayed. 2024. [Mixture of experts.](#)

- Peng Jin, Bo Zhu, Li Yuan, and Shuicheng Yan. 2024. [Moe++: Accelerating mixture-of-experts methods with zero-computation experts](#).
- Zhenzhong Lan, Mingdao Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*.
- Dmitry Lepikhin, Hyoukseo Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2020. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*.
- Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Timothy Baldwin. 2024a. [CMMLU: measuring massive multitask language understanding in chinese](#). In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, volume ACL 2024 of *Findings of ACL*, pages 11260–11285. Association for Computational Linguistics.
- Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Yitzhak Gadre, Hritik Bansal, Etash Kumar Guha, Sedrick Scott Keh, Kushal Arora, Saurabh Garg, Rui Xin, Niklas Muennighoff, Reinhard Heckel, Jean Mercat, Mayee F. Chen, Suchin Gururangan, Mitchell Wortsman, Alon Albalak, Yonatan Bitton, Marianna Nezhurina, Amro Abbas, Cheng-Yu Hsieh, Dhruba Ghosh, Josh Gardner, Maciej Kilian, Hanlin Zhang, Rulin Shao, Sarah M. Pratt, Sunny Sanyal, Gabriel Ilharco, Giannis Daras, Kalyani Marathe, Aaron Gokaslan, Jieyu Zhang, Khyathi Raghavi Chandu, Thao Nguyen, Igor Vasiljevic, Sham M. Kakade, Shuran Song, Sujay Sanghavi, Fartash Faghri, Sewoong Oh, Luke Zettlemoyer, Kyle Lo, Alaaeldin El-Nouby, Hadi Pouransari, Alexander Toshev, Stephanie Wang, Dirk Groeneveld, Luca Soldaini, Pang Wei Koh, Jenia Jitsev, Thomas Kollar, Alex Dimakis, Yair Carmon, Achal Dave, Ludwig Schmidt, and Vaishaal Shankar. 2024b. [Datacomp-lm: In search of the next generation of training sets for language models](#). In *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.
- Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. [Program induction by rationale generation: Learning to solve and explain algebraic word problems](#). In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 158–167. Association for Computational Linguistics.
- Niklas Muennighoff et al. 2025. Olmoe: Open mixture-of-experts language models. *arXiv preprint arXiv:2501.00662*.
- Van Bach Nguyen, Christin Seifert, and Jörg Schlötterer. 2024. [CEval: A benchmark for evaluating counterfactual text generation](#). In *Proceedings of the 17th International Natural Language Generation Conference*, pages 55–69, Tokyo, Japan. Association for Computational Linguistics.
- Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 157–163. Association for Computational Linguistics.
- Zhihong Shao, Damai Dai, Daya Guo, Bo Liu (Benjamin Liu), Zihan Wang, and Huajian Xin. 2024. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *ArXiv*, abs/2405.04434.
- Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*.
- Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. 2023. [Challenging BIG-Bench tasks and whether chain-of-thought can solve them](#). In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, volume ACL 2023 of *Findings of ACL*, pages 13003–13051. Association for Computational Linguistics.
- Xiaomi AI Team. 2026. Mimo-v2-flash: Ultra-sparse moe for agentic workflows. *Technical Report*.
- Yudong Wang, Zixuan Fu, Jie Cai, Peijun Tang, Hongya Lyu, Yewei Fang, Zhi Zheng, Jie Zhou, Guoyang Zeng, Chaojun Xiao, Xu Han, and Zhiyuan Liu. 2025. [Ultra-fineweb: Efficient data filtering and verification for high-quality LLM training data](#). *CoRR*, abs/2505.05427.
- Tiansheng Wen, Yifei Wang, Aosong Feng, Long Ma, Xinyang Liu, Yifan Wang, Lixuan Guo, Bo Chen, Stefanie Jegelka, and Chenyu You. 2025. [Route experts by sequence, not by token](#).
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiayi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang,

Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. 2025. [Qwen3 technical report](#).

Yuanhang Yang, Shiyi Qi, Wenchao Gu, Chaozheng Wang, Cuiyun Gao, and Zenglin Xu. 2024. [XMoE: Sparse models with fine-grained and adaptive expert selection](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 11664–11674, Bangkok, Thailand. Association for Computational Linguistics.

Sha Yuan, Hanyu Zhao, Zhengxiao Du, Ning Ding, Xiao Liu, Yukuo Cen, Xu Huang, et al. 2021. Wudaocorpora: A super large-scale chinese corpora for pre-training language models. *AI Open*, 2:65–68.

Longfei Yun, Yonghao Zhuang, Yao Fu, Eric P. Xing, and Hao Zhang. 2024. Toward inference-optimal mixture-of-expert large language models. *ArXiv*, abs/2404.02852.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. [Instruction-following evaluation for large language models](#). *CoRR*, abs/2311.07911.

A Related Work

A.1 Mixture-of-Experts Transformers

Mixture-of-Experts (MoE) has emerged as the leading paradigm for scaling Transformer capacity while maintaining constant computational costs per token. Foundational works established the sparsely-gated MoE layer (Shazeer et al., 2017) and demonstrated its scalability to hundreds of billions of parameters through automatic sharding (Lepikhin et al., 2020). The Switch Transformer (Fedus et al., 2022) further optimized this via Top-1 routing, significantly improving training efficiency over dense baselines. Recent innovations have focused on refining routing granularity and expert specialization. DeepSeekMoE (Dai et al., 2024) and DeepSeek-V2 (Shao et al., 2024) introduced fine-grained expert segmentation and shared experts to better capture common knowledge, while Mixtral (Jiang et al., 2024) validated the efficacy of Top-2 routing for open-weight models. Further architectural refinements include OL-MoE (Muennighoff et al., 2025), which provides open-source scaling insights, and MoE++ (Jin et al., 2024), which utilizes zero-computation experts to reduce FLOPs. Efficiency is further pushed by X-MoE (Yang et al., 2024) through threshold-based routing and Seqtopk (Wen et al., 2025), which shifts from token-level to sequence-level routing to minimize communication overhead. Most recently, MiMo-V2-Flash (Team, 2026) has optimized ultra-sparse designs specifically for agentic workflows. Despite these diversities in routing and granularity, existing MoE architectures share a fundamental constraint: layer-wise isolation. Experts are strictly confined to their respective depths, preventing the exploitation of cross-layer semantic redundancies. Our proposed CS-MoE addresses this limitation by enabling longitudinal expert reusability, thereby maximizing parameter utility across the entire network hierarchy.

A.2 Parameter Sharing in Transformers

Parameter sharing has long served as a vital mechanism for enhancing model generalization and efficiency, with early research focusing on weight tying between embedding and output layers (Press and Wolf, 2017) in Transformers. This evolved into the full cross-layer parameter sharing introduced by ALBERT (Lan et al., 2020), though its reliance on uniform sharing at every depth inherently limits the model’s ability to learn specialized

representations. While subsequent investigations in DeBERTa (He et al., 2021) and PaLM (Chowdhery et al., 2023) identified functional similarities across layers, and Universal Transformers (Dehghani et al., 2019) proposed recurrent sharing, these approaches often introduce sequential bottlenecks that compromise parallelization. Within the Mixture-of-Experts (MoE) domain, most architectures maintain strict layer-wise isolation of experts (Shazeer et al., 2017; Lepikhin et al., 2020; Fedus et al., 2022). DeepSeekMoE (Dai et al., 2024) attempted to bridge this gap with "shared experts," yet these remain restricted to intra-layer application and do not exploit the longitudinal redundancies observed in recent analyses like MoE++ (Jin et al., 2024). CS-MoE addresses these fundamental constraints by implementing a dual-tier hierarchy that balances independent, depth-specific experts with a global shared pool. Unlike the rigid uniformity of ALBERT or the static recurrence of Universal Transformers, CS-MoE utilizes per-token dynamic routing to enable genuine inter-layer allocation, thereby achieving superior parameter efficiency while preserving the essential hierarchical capacity of deep Transformer networks.

B Pilot Study

B.1 Implementation Details

We conducted our analysis using a pre-trained Qwen3-MoE-4B model, which features 36 Transformer layers and 4 experts per layer ($N = 4$, $\text{Top-}K = 1$). The study was performed on a held-out validation set of 50,000 tokens sampled from the DCLM corpus to ensure statistical significance in activation patterns.

Representational Similarity Analysis. To quantify the functional overlap between experts without being misled by the permutation invariance of neural weights, we employed Centered Kernel Alignment (CKA). Unlike Cosine similarity, CKA is invariant to orthogonal transformations and isotropic scaling, making it the standard for comparing internal representations.

For two expert weight matrices $W_i^{(l)}$ and $W_j^{(l')}$ from layers l and l' , we computed the linear CKA score as follows:

$$CKA(K, L) = \frac{HSIC(K, L)}{\sqrt{HSIC(K, K)HSIC(L, L)}}$$

where $K = XX^T$ and $L = YY^T$ are the Gram matrices of the activations produced by the respective

Table 4: Validation perplexity increase ($+\Delta$ PPL) under targeted CKA vs. random expert substitutions.

Substitution Type	Direction	$+\Delta$ PPL (ESR=5%)	$+\Delta$ PPL (ESR=20%)
Targeted (CKA)	Shallow $\rightarrow$ Deep	+1.22	+3.45
	Deep $\rightarrow$ Shallow	+1.54	+3.98
Random Swap	Shallow $\leftrightarrow$ Deep	+4.20	+12.80
Random Control	Arbitrary Layer Swap	+4.15	+15.30

experts, and $HSIC$ is the Hilbert-Schmidt Independence Criterion. A high CKA score (> 0.85) between a shallow-layer expert and a deep-layer expert served as our primary indicator of longitudinal redundancy.

Zero-Shot Expert Substitution Protocol. To verify the interchangeability of experts, we implemented a greedy substitution algorithm:

1. **Selection:** For each expert in the target "Deep" layers (layers 24-32), we identified a "Source" expert from the "Shallow" layers (layers 1-8) that maximized the CKA score.
2. **Replacement:** We performed a direct parameter swap, replacing the weights of the Deep expert with the weights of its Shallow counterpart.
3. **Inference:** We measured the change in model perplexity on the validation set without any gradient updates or fine-tuning.
4. **Baseline:** As a control, we performed random substitutions of equal magnitude.

The moderate increase in perplexity (under 15 within the acceptable region) during targeted substitution, compared to a sharp divergence in the random baseline, empirically justifies the feasibility of the shared expert pool proposed in CS-MoE.

B.2 Extension: Bidirectional CKA and Random Swaps

To rigorously test cross-layer functional equivalence, we expanded the pilot study on Qwen3-MoE-4B to include bidirectional substitutions (Deep $\rightarrow$ Shallow) as well as randomized control swaps under Expert Sharing Ratios (ESR) of 5% and 20%. As shown in Table 4, targeted CKA swaps in both directions incur minimal perplexity increases, while random permutations lead to severe degradation. This demonstrates that functional redundancy is symmetric and structured across network depths.

C Training Details & Hyperparameters

All standard pre-training runs were conducted using customized Megatron-LM on an $8 \times$ NVIDIA

Table 5: Detailed model hyperparameters. *: Top- K activated experts include independent experts of each layer.

Model Type	Total Params	Activated Params	Layers (L)	d_{ffn}	d_{exp}	N_{indep}	Shared Pool (M)	Top- K *
Dense	0.6B / 1.7B / 4B / 8B	Same	16 - 36	3,072 - 12,288	-	-	-	-
CS-MoE 0.6B	0.6B	0.6B / 0.9B / 1.7B	28	3,072	768	1	84	4 / 10 / 21
CS-MoE 1.7B	1.7B	1.7B / 4B	28	6,144	1,536	1	84	4 / 13
CS-MoE 4B	4B	4B / 8B	36	9,728	2,432	1	108	4 / 10
CS-MoE 8B	8B	4B	36	9,728	2,432	1	324	4
CS-MoE 8B	8B	8B	36	12,288	3,072	1	324	4
CS-MoE 12B	12B	4B	36	9,728	2,432	1	540	4

Table 6: Unified Training Hyperparameters.

Hyperparameters	Settings
Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.95, \epsilon = 1e-8$)
Peak Learning Rate	3e-4
Weight Decay	0.01
Batch Size	512 (effective token count $\approx 1M$)
Warmup Steps	1,000
Total Steps	12,500 (Base) / 4.5T tokens (Scaled)
LR Schedule	Cosine Annealing to $0.1 \times LR_{max}$
Context Length	2,048
Aux Balancing α	0.01

Table 7: Final pre-training perplexity corresponding to Figure 3 (CS-MoE vs. Dense across scales).

Model Scale	Dense Baseline	CS-MoE (Ours)	Reduction (%) $\downarrow$
0.6B Scale	13.62	13.48	1.03%
1.7B Scale	11.30	11.12	1.59%
4B Scale	10.45	10.08	3.54%
8B Scale	9.75	9.55	2.05%

Table 8: Final pre-training perplexity corresponding to Figure 4 (CS-MoE with varying activation counts K).

Model Scale	Activation Configuration	Final Perplexity (PPL)
0.6B Scale	0.6B-A0.6B	13.48
	0.6B-A0.9B	13.18
	0.6B-A1.7B	13.11
1.7B Scale	1.7B-A1.7B	11.12
	1.7B-A4B	10.82
4B Scale	4B-A4B	10.08
	4B-A8B	9.95

H200 GPU node. Hyperparameters are detailed in Table 6. For the scaling run on the 4.5T-token corpus, gradient accumulation was scaled across 32 nodes with identical learning rate schedules.

Table 5 details the structural hyperparameters across all model scales evaluated.

D More Results

D.1 Exact Numerical PPL Tables

To complement the curves presented in Figures 3, 4, and 5, we report the exact final pre-training perplexity (PPL) values below.

Table 9: Final pre-training perplexity corresponding to Figure 5 (CS-MoE vs. Standard MoE under double-aligned total and active parameters).

Configuration	Standard MoE	CS-MoE (Ours)	Reduction (%) $\downarrow$
8B-A4B	9.70	9.65	0.52%
12B-A4B	9.40	9.19	2.23%

Table 10: Mean and standard deviation over 3 independent training runs at 4B scale.

Model	Pre-train PPL	C-Eval (%)	AQuA (%)
4B Dense Baseline	10.25 $\pm$ 0.04	40.8 $\pm$ 0.3	39.0 $\pm$ 0.5
4B CS-MoE (Ours)	10.06 $\pm$ 0.03	42.3 $\pm$ 0.2	40.5 $\pm$ 0.4

D.2 Statistical Significance and Seed Variance

To ensure observed performance gains are statistically robust, we conducted 3 independent training runs with distinct random seeds at the 4B scale (12.5k steps). As shown in Table 10, CS-MoE’s performance gains consistently exceed the standard deviations across all evaluation metrics.

D.3 Hardware Profiling & Latency Analysis

We profiled the computational and memory overhead on an 8 $\times$ NVIDIA H200 (141GB HBM3e) GPU cluster.

Routing Computational Cost For hidden dimension $d = 4096$ and shared pool $M = 324$, the gating projection matrix has $4096 \times 324 \approx 1.3M$ parameters. Computed at each layer, gating operations account for less than 0.1% of the total FLOPs per forward pass.

Inference Latency with Expert Caching Because adjacent layers frequently access overlapping subsets of shared experts, active weights can be retained in GPU L2/SRAM cache across sequential layer executions. Table 11 reports latency measurements at batch size 1.

With cache optimization, CS-MoE approaches the inference speed of Dense models and outperforms standard MoEs due to a 45% reduction in unique memory traffic per token.

Table 11: Real-world latency profiling at 8B model scale on $8 \times$ NVIDIA H200 GPUs.

Model Configuration	Prefill (ms)	Decoding (ms/tok)
Dense 8B	12.5	14.2
Standard Sparse MoE 8B	14.1	16.5
CS-MoE 8B (Default)	13.8	15.8
CS-MoE 8B (with Cache Optimization)	13.1	14.9

D.4 Expert Routing Dynamics across Layers

We analyzed the routing patterns across all $L = 36$ layers of the fully-trained 8B CS-MoE model:

- **Layer-wise Active Overlap:** The overlap ratio of the top-10% most frequently activated shared experts between adjacent layers (l and $l + 1$) is **68.4%**, gradually decaying to **42.1%** between distant layers (l and $l + 5$). This confirms smooth longitudinal semantic transitions.
- **Expert Specialization (Gini Index):** The average Gini coefficient of expert activation frequencies across layers is **0.28** (where 0 indicates perfectly uniform global usage and 1 indicates single-layer isolation). This confirms that shared experts are globally utilized rather than confined to specific depths.