

Late Breaking Results: Structured Expert Routing for Efficient MoE Inference under GPU–CPU Orchestration

Yixiao Chen^{1,4*}, Arman Akbari^{1*}, Arash Akbari¹, Zhendong Mi², Enfu Nan¹, Xiaowei Lin³,
Weiwei Chen⁴, Shaoyi Huang², Hao Wang², Pu Zhao^{1,†}, Yanzhi Wang^{1,4}

¹Northeastern University ²Stevens Institute of Technology ³ETH Zurich ⁴EmbodysX Inc.

Abstract

Hybrid GPU–CPU deployment of large Mixture-of-Experts (MoE) models is often latency-bound by CPU–side expert execution and cross-device orchestration overhead. To overcome this limitation, we propose a structured expert routing strategy, implemented via asymmetric expert skipping, which reduces expert activation in latency-critical layers while applying lightweight magnitude compensation and norm calibration to preserve model accuracy. Experiments on quantized Qwen3 (30B and 235B) models show that our method improves end-to-end generation throughput by up to 60% while achieving 3.6% higher accuracy than uniform expert reduction under the same compute budget.

1 Introduction

The scaling of Large Language Models (LLMs) has been significantly accelerated by the adoption of Mixture-of-Experts (MoE) architectures [11]. By sparsely activating a subset of feed-forward networks for each token, MoE models decouple parameter count from inference computation, enabling massive model capacities. However, deploying such large-scale architectures in memory-constrained systems remains challenging. To address this issue, prior works have explored both algorithmic compression techniques [8, 12, 14] and hardware-level hybrid deployment strategies [2, 9, 13]. The latter leverages the cost-effective memory capacity of CPU/DRAM alongside the high computational bandwidth of GPU/VRAM. In such hybrid systems, inference latency is largely dominated by CPU–side expert execution and GPU–CPU coordination overhead.

While dynamic or input-adaptive routing can improve expert utilization, such strategies introduce token-level variability in expert activation, leading to unpredictable CPU workloads and irregular cross-device communication in GPU–CPU systems. Motivated by this observation, we propose a structured expert routing framework tailored for GPU–CPU orchestration as demonstrated in Figure 1. Specifically, we selectively reduce the active expert count (K') in targeted layers to decrease CPU–side expert execution and overall orchestration latency, and mitigate the induced accuracy degradation through unnormalized residual scaling and parameter-efficient tuning of the pre-expert normalization modules. Experiments on quantized Qwen3 models (30B and 235B) demonstrate that our strategy improves generation throughput by 60% while maintaining competitive accuracy on standard reasoning benchmarks.

*These authors contributed equally to this work. † Corresponding author.



This work is licensed under a Creative Commons Attribution 4.0 International License. DAC '26, Long Beach, CA, USA

© 2026 Copyright held by the owner/author(s).

ACM ISBN 979-8-4007-2254-7/2026/07

<https://doi.org/10.1145/3770743.3807684>

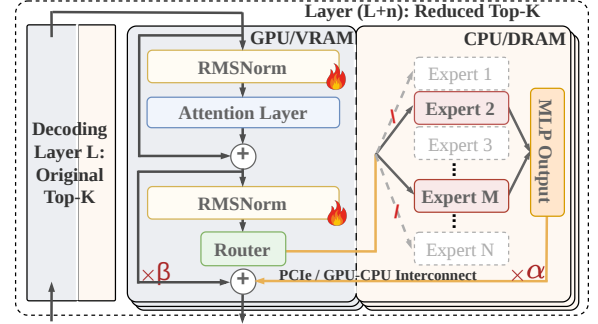


Figure 1: Asymmetric expert skipping strategy for structured expert routing under GPU–CPU orchestration.

2 Asymmetric Expert Skipping

Let $X_{attn} \in \mathbb{R}^{B \times S \times d}$ denote the hidden state after the self-attention sub-layer and its residual connection. The input to the MoE block is first normalized as $H_{mlp} = \text{RMSNorm}(X_{attn})$. A routing network W_r computes a routing probability distribution over N expert networks $\{E_1, E_2, \dots, E_N\}$ via $P = \text{Softmax}(W_r H_{mlp})$. Let Λ_K denote the indices of the top- K probabilities. Standard MoE implementations renormalize the selected top- K routing probabilities to sum to 1. The final layer output is the probability-weighted sum of the activated experts added back to the residual stream:

$$X^{(l+1)} = X_{attn} + \sum_{i \in \Lambda_K} \left(\frac{P_i}{\sum_{j \in \Lambda_K} P_j} \right) E_i(H_{mlp}) \quad (1)$$

Under GPU–CPU orchestration, slow expert computation on the CPU creates a severe inference bottleneck. To alleviate this without introducing the latency jitter of dynamic routing or the capacity degradation of uniform reduction, we propose a structured expert routing strategy implemented via asymmetric expert skipping. This structured strategy partitions the network’s MoE layers into two distinct subsets: a full-capacity set $\mathcal{L}_{\text{base}}$ operating at the original K , and a reduced-capacity set $\mathcal{L}_{\text{reduced}}$ statically operating at $K' < K$.

2.1 Global Scale Compensation

Dropping experts from K to K' in some layers discards the probability mass associated with the tail of the routing distribution. If the remaining probabilities are renormalized to sum to 1, the contribution of the retained experts can be artificially amplified, causing severe distribution shifts in the residual stream. To compensate for this effect, we introduce global scaling coefficients α and β applied to the MoE branch and residual branch, respectively. For $l \in \mathcal{L}_{\text{reduced}}$, the residual update becomes:

$$X^{(l+1)} = \beta X_{attn} + \alpha \sum_{i \in \Lambda_{K'}} \left(\frac{P_i}{\sum_{j \in \Lambda_{K'}} P_j} \right) E_i(H_{mlp}). \quad (2)$$

Model	Configuration	$\bar{K}$	PPL ↓	MMLU	GSM8K	IFEval	MBPP	Avg. ↑	TPS ↑
Qwen3-30B-A3B 4-bit[4]	Original Setting	8	7.36	79.95	87.49	70.50	75.60	78.39	29.45
	Uniform Expert Reduction	6	7.78	79.26	86.28	67.39	72.80	76.43	39.27
	Uniform Expert Reduction	5	8.10	76.13	84.00	64.15	69.80	73.52	45.10
	Uniform Expert Reduction	4	8.62	73.65	77.94	61.03	60.80	68.36	51.84
	Proposed (Structured)	5	7.69	78.24	86.66	69.66	74.20	77.19	43.14
Qwen3-235B-A22B 4-bit[4]	Original Setting	8	4.31	87.41	96.66	93.88	85.40	90.84	7.31
	Uniform Expert Reduction	6	4.83	86.42	96.21	92.21	81.00	88.98	9.54
	Uniform Expert Reduction	5	5.05	84.72	93.18	90.89	73.40	85.55	11.54
	Uniform Expert Reduction	4	5.61	83.54	88.63	88.25	37.00	74.36	13.07
	Proposed (Structured)	5	4.70	86.11	95.68	92.57	82.60	89.24	11.68

Table 1: Performance and latency of Qwen3 MoE models. Our structured expert routing ($\bar{K} = 5$) surpasses the uniform $K = 6$ baseline across reasoning tasks while achieving up to 60% higher throughput than the original model.

The coefficients α and β are determined using a small calibration set. We match the cumulative squared activation magnitude of the reduced configuration to that of the original K setting via a lightweight greedy search over α and β . This calibration is performed once offline and introduces no additional inference overhead.

2.2 Parameter-efficient Selective Norm-Tuning

While α and β correct global magnitude shifts, dropping specific experts alters the variance of the hidden states on a per-channel basis. To recover from this fine-grained variance shift, we propose a parameter-efficient calibration phase. Each transformer block contains two RMSNorm layers: one preceding the self-attention sub-layer and one preceding the MoE MLP block. We freeze all network weights and exclusively fine-tune only the scale vectors $\gamma \in \mathbb{R}^d$ of both normalization layers for modified blocks ($l \in \mathcal{L}_{\text{reduced}}$):

$$\text{RMSNorm}(X) = \frac{X}{\sqrt{\frac{1}{d} \sum_{j=1}^d X_j^2 + \epsilon}} \odot \gamma \quad (3)$$

The trainable parameter set is strictly limited to $\Theta = \{\gamma_{\text{attn}}^{(l)}, \gamma_{\text{mlp}}^{(l)} \mid l \in \mathcal{L}_{\text{reduced}}\}$. We optimize Θ using the standard causal language modeling objective over a small calibration dataset.

3 Experiments

We applied our method on quantized [4] Qwen3-30B-A3B-Instruct-2507 [11] and Qwen3-235B-A22B-Instruct-2507 [11]. All experiments were conducted on a server equipped with dual AMD EPYC 7513 CPUs (64 cores total) and NVIDIA L40S GPU with 48 GB memory. The GPU–CPU interconnect uses PCIe Gen4 x16. All experts are offloaded to the CPU to emulate hybrid GPU–CPU deployment.

We set $K' = 4$, $\alpha = 0.8$, and $\beta = 1.0$ in 75% of layers (with an average $\bar{K} = 5$) and perform Norm-Tuning using 12k samples in total from RedPajamaV2 [10] and StarCoder [6]. We benchmark against the original $K = 8$ setting and uniform expert reduction baselines ($K = 4, 5, 6$). Perplexity and task accuracy are measured across WikiText [7], MMLU [5], GSM8K [3], IFEval [15], and MBPP [1].

Table 1 summarizes the perplexity, task accuracy, and throughput (tokens/s) under different expert configurations. Relative to the $K = 8$ baseline, our method improves TPS from 29.45 to 43.14 on the 30B model (46%) and from 7.31 to 11.68 on the 235B model

(60%), with negligible degradation in average accuracy. Notably, even with an average of only 5 active experts, our structured routing design surpasses the uniform $K = 6$ setting on both model scales, highlighting that structured expert routing is more effective than uniform reduction under GPU–CPU orchestration.

Conclusion

We introduce Asymmetric Expert Skipping to enable efficient MoE inference under GPU–CPU orchestration. By reducing the active expert count (with an average $\bar{K} = 5$) and recovering accuracy via lightweight magnitude and norm calibration, our method increases throughput by over 60% with negligible degradation compared to original $K = 8$ models. Furthermore, it yields a 3.6% average accuracy improvement over uniform $K = 5$ baselines, outperforming even heavier $K = 6$ configurations.

Acknowledgments

This research is supported in part by grants NSF CMMI 2402438.

References

- [1] Jacob Austin et al. 2021. Program Synthesis with Large Language Models. *arXiv preprint arXiv:2108.07732* (2021).
- [2] Yixiao Chen et al. 2025. Collaborative Compression for Large-Scale MoE Deployment on Edge. *arXiv:2509.25689* [cs.AI]
- [3] Karl Cobbe et al. 2021. Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168* (2021).
- [4] Georgi Gerganov and contributors. 2023. llama.cpp: Inference of LLaMA model in pure C/C++. <https://github.com/ggml-org/llama.cpp>.
- [5] Dan Hendrycks et al. 2021. Measuring Massive Multitask Language Understanding. *ICLR* (2021).
- [6] Raymond Li et al. 2023. StarCoder: may the source be with you! *arXiv:2305.06161*
- [7] Stephen Merity et al. 2016. Pointer Sentinel Mixture Models. *arXiv:1609.07843*
- [8] Xuan Shen et al. 2024. Search for Efficient Large Language Models. In *Advances in Neural Information Processing Systems (NeurIPS) 2024*.
- [9] Xuan Shen et al. 2025. Sparse Learning for SSMs on Mobile. In *ICLR* (2025).
- [10] Maurice Weber et al. 2024. RedPajama: an Open Dataset for Training Large Language Models. *arXiv:2411.12372* [cs.CL]
- [11] An Yang et al. 2025. Qwen3 Technical Report. *arXiv:2505.09388* [cs.CL]
- [12] Zheng Zhan et al. 2024. Exploring Token Pruning in Vision State Space Models. In *Advances in Neural Information Processing Systems (NeurIPS) 2024*.
- [13] Zheng Zhan et al. 2024. Fast and Memory-Efficient Video Diffusion Using Streamlined Inference. In *NeurIPS 2024*.
- [14] Pu Zhao et al. 2024. Pruning Foundation Models for High Accuracy without Retraining. In *EMNLP Findings 2024*.
- [15] Jeffrey Zhou et al. 2023. Instruction-Following Evaluation for Large Language Models. *arXiv:2311.07911* [cs.CL]