ICML 2026 Spotlight

Attribution-Guided and Coverage-Maximized Pruning for Structural MoE Compression

Authors: Yifu Ding[12], Jiacheng Wang[14], Ge Yang[14], Yongcheng Jing[3], Jinyang Guo[14], Xianglong Liu[12], Dacheng Tao[3]

Affiliations: [1]State Key Laboratory of Complex & Critical Software Environment, Beihang University, [2]School of Computer Science and Engineering, Beihang University, [3]Nanyang Technological University, [4]School of Artificial Intelligence, Beihang University

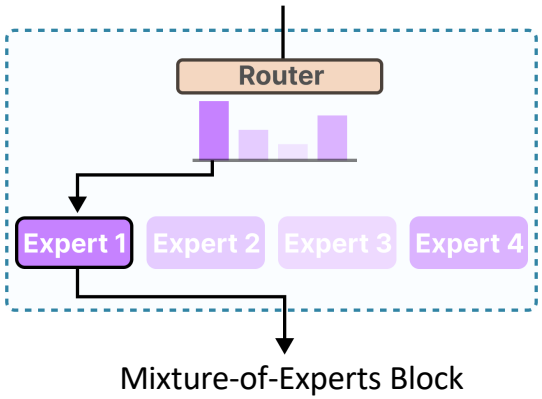
北京航空航天大學 BEIHANG UNIVERSITY

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Background

MoE models are becoming increasingly popular backbones for scaling language models.

- They provide high parameter capacity while maintaining manageable computation.
- This is achieved by activating only a subset of experts for each token.

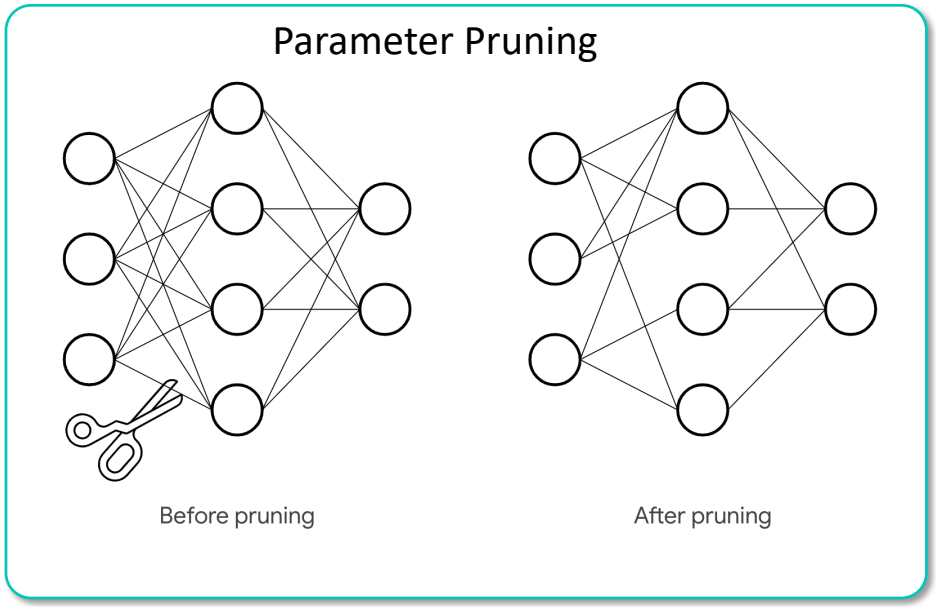


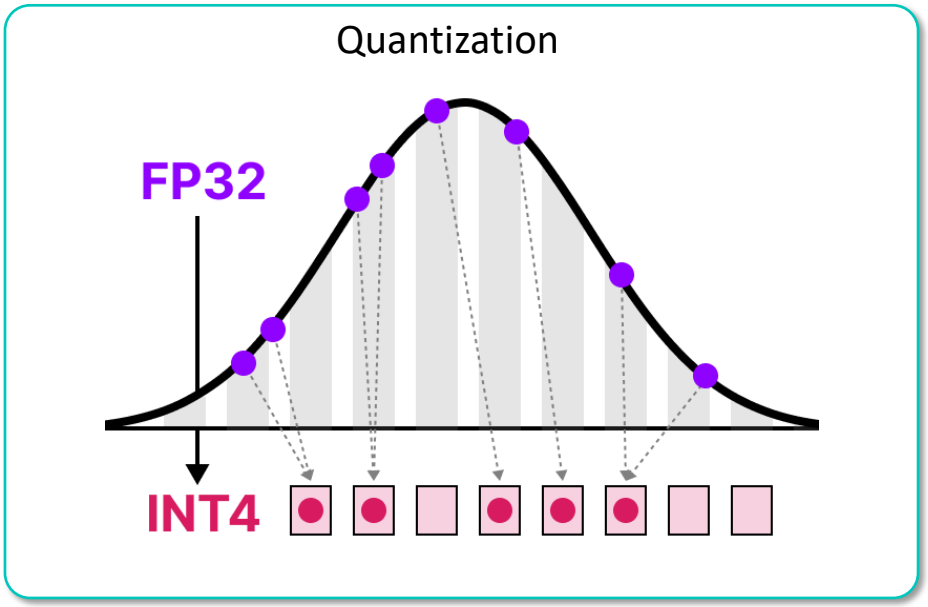
Recent Large MoEs

- DeepSeek-V3: 671B Total, 37B Active, 256 Experts
- Kimi K2.5: 1T Total, 32B Active, 384 Experts
- Llama 4 Maverick: 400B Total, 17B Active, 128 Experts
- Qwen3-235B-A22B: 235B Total, 22B Active, 128 Experts
- Snowflake Arctic: 480B Total, 17B Active, 128 Experts

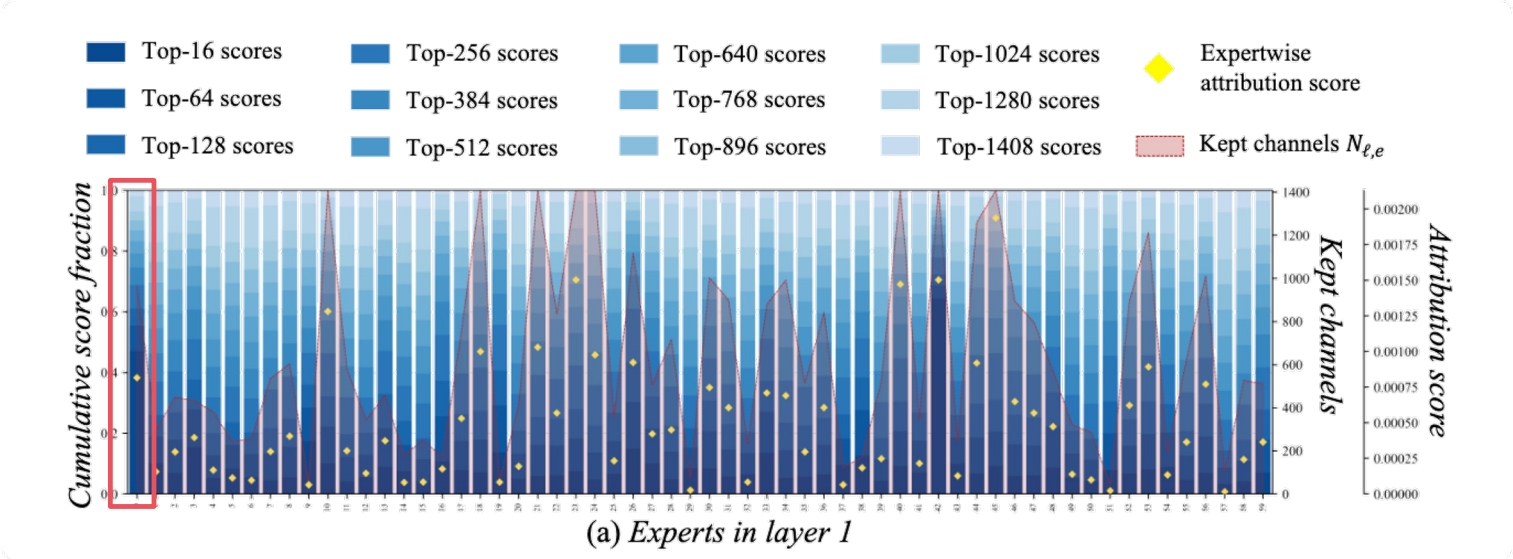
Background

Model compression methods





Problem Motivation

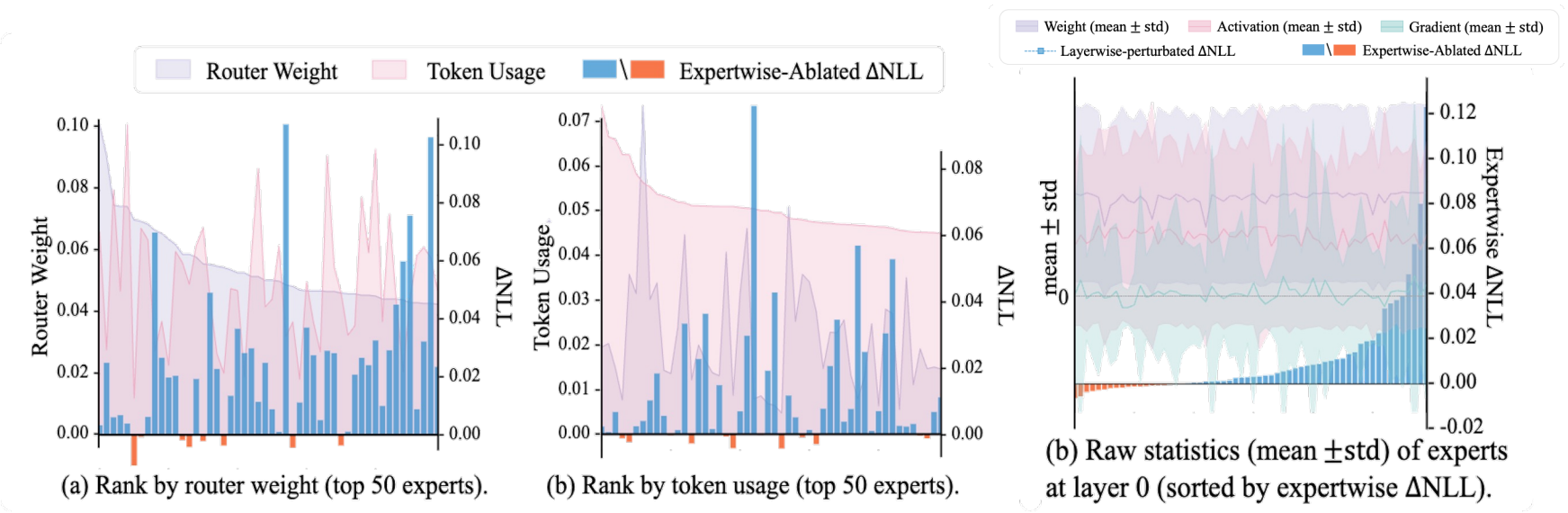


Observation 1:

Within a single MoE expert, some channels exhibit larger activation magnitudes, indicating that information is concentrated in a small subset of channels.

👉 This suggests the possibility of more fine-grained structured pruning.

Problem Motivation



(a) Rank by router weight (top 50 experts). (b) Rank by token usage (top 50 experts). (b) Raw statistics (mean ±std) of experts at layer 0 (sorted by expertwise ΔNLL).

Observation 2:

Existing metrics, such as router logits, token count, weight, activation, and gradients, show no clear correlation with the actual expert-ablated loss.

👉 Therefore, they cannot accurately reflect expert importance or contribution.

Method Overview

Proposed Methods

1. Coverage-Maximized Budget Allocation 👉 A channel allocation strategy for structured pruning
2. Attribution-Guided Loss Approximation 👉 An accurate and efficient estimation of expert contribution
3. Alignment-Aware Redistribution 👉 Compatible with mainstream quantization methods

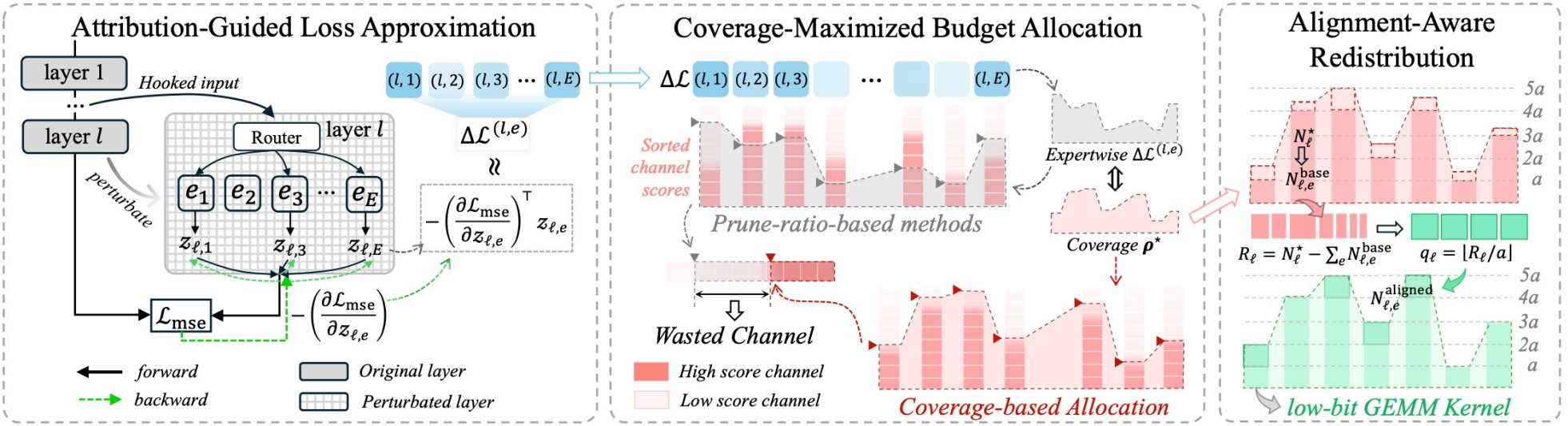


Figure 5. The overview of Attribution-Guided & Coverage-Maximized Expert-wise Pruning framework for MoE models.

Method1: Coverage-Maximized Budget Allocation

Goal: **Allocate fewer channels to experts whose activation information is more concentrated.**

Key Idea: **Use each expert's contribution to guide channel allocation:**

- **Number of channels ∝ expert contribution** ✗
- **Cumulative channel score ∝ expert contribution** ✓

Preparation

Step 1. Collect scores for each expert and each channel.

$s_{\ell,c}$	ARC-c	GSM8K	HumanEval
Weight (Equation (32))	38.0	25.9	18.9
Activation (Equation (33))	**40.0**	**58.2**	**30.5**
Gradient Saliency (Equation (35))	39.0	51.7	26.8
Weight×Activation (Equation (34))	38.2	33.7	28.7
Activation×Gradient (Equation (36))	38.7	46.3	26.2
SNIP First-order Sensitivity (Weight×Gradient) (Equation (37))	39.2	41.5	24.4

In our experiments, we use the sum of absolute channel activations, which is more suitable for the evaluated tasks but has not been extensively validated.

Step 2. Compute the prefix sum and total sum of channel scores.

Prefix Sum $S_\ell(n) = \sum_{i=1}^{n} s_{\ell,(i)}, \quad 0 \le n \le |\mathcal{C}_\ell|, \quad S_\ell(0) = 0$

Total Sum: $S_\ell^{tot} = \sum_{c \in \mathcal{C}_\ell} s_{\ell,c}$

This enables efficient computation of the score ratio covered by the first n channels: $\rho_\ell(n) = \frac{S_\ell(n)}{S_\ell^{tot}}$

Step 3. *Collect layer-wise loss as the layer importance score for inter-layer allocation.*
Collect expert-wise importance as the expert importance score for intra-layer allocation.
👉*Details are provided in Method 2.*

Method1: Coverage-Maximized Budget Allocation

Goal: **Allocate fewer channels to experts whose activation information is more concentrated.**

Key Idea: **Use each expert's contribution to guide channel allocation:**

- **Number of channels ∝ expert contribution** ✗
- **Cumulative channel score ∝ expert contribution** ✓

Binary Search: Maximize the global retained-channel score ratio $\rho_\ell(n)$

Algorithm 1 Coverage-Maximized Allocation Search

Input: Score allocation weights $\boldsymbol{\phi} \in \mathbb{R}_+^{|\mathcal{G}|}$; prefix sums $\{\mathcal{S}_g(n)\}_{g\in\mathcal{G}}$; total scores $\{\mathrm{S}_g^{tot}\}_{g\in\mathcal{G}}$; channel budget N_{budget}; total channels N^{tot}; tolerance ε.
Output: Channel budgets $\{N_g^\star\}_{g\in\mathcal{G}}$
$\alpha_{\min} \leftarrow 0, \alpha_{\max} \leftarrow 1$
while $\alpha_{\min} < \alpha_{\max}$ **do**
  $\alpha \leftarrow (\alpha_{\min} + \alpha_{\max})/2$
  $\boldsymbol{\rho} \leftarrow \min(\alpha\,\boldsymbol{\phi}, 1)$
  $N(\boldsymbol{\rho}) \leftarrow \sum_{g\in\mathcal{G}} \min\{n \mid \mathcal{S}_g(n) \geq \rho_g(\alpha)\,\mathrm{S}_g^{tot}\}$
  if $|N(\boldsymbol{\rho}) - N_{\text{budget}}| \leq \varepsilon\, N^{tot}$ **then**
    $N_g^\star \leftarrow \min\{n \mid \mathcal{S}_g(n) \geq \rho_g(\alpha)\,\mathrm{S}_g^{tot}\}, \forall g \in \mathcal{G}$
    break
  end if
  if $N(\boldsymbol{\rho}(\alpha)) > N_{\text{budget}}$ **then**
    $\alpha_{\max} \leftarrow \alpha$
  else
    $\alpha_{\min} \leftarrow \alpha$
  end if
end while
return $\{N_g^\star\}_{g\in\mathcal{G}}$

Algorithm Explanation

#L3: Initialize the search boundaries $\alpha_{\min}, \alpha_{\max}$

#L5: Use α as the binary search pivot.

#L6: $\boldsymbol{\rho}$: the current retained-score ratio, represented as a 1-D vector containing the individual threshold for each expert in the layer.

#L7: Compute the minimum total number of channels $N(\boldsymbol{\rho})$ required to satisfy the current $\boldsymbol{\rho}$, using the prefix sums from Step 2 for fast lookup.

#L8: Check whether the planned total number of channels $N(\boldsymbol{\rho})$ satisfies the overall budget N_{budget}.

A small toleranceε is allowed because the number of channels must be integer-valued, so it may not exactly match the budget.

#L9-10: If the overall budget is satisfied, terminate the search.

#L12-16: If the result is above or below the budget, update the binary search boundaries accordingly.

Method1: Coverage-Maximized Budget Allocation

Algorithm Efficiency of Binary Search

Computational Complexity:

Since prefix sums are precomputed, $S_g(n)$ is monotonic non-decreasing with respect to n. Meanwhile, $N(\rho(\alpha))$ is monotonic non-decreasing with respect to α. Therefore, the following expression can be evaluated in $O(\log |\mathcal{C}_\ell|)$ time, where $\mathcal{C}_\ell$ is the total set of channels:

$$N(\rho) = \sum_{g \in \mathcal{G}} N_g(\rho_g) = \sum_{g \in \mathcal{G}} \min\{n \mid S_g(n) \geq \rho_g S_g^{tot}\}$$

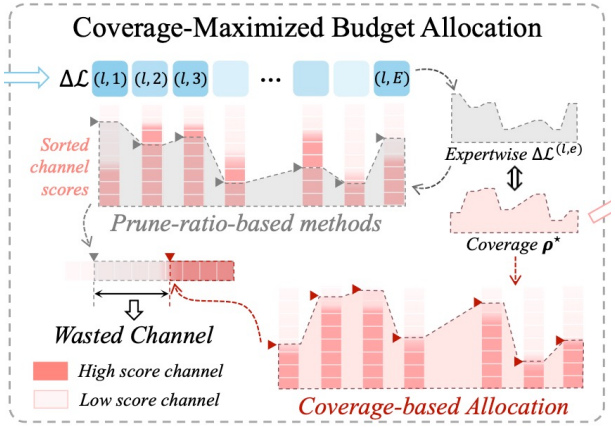


Illustration of redistributing channels

Runtime Statistics:

On a 30B MoE model, the total runtime is around 10 seconds.

Stage	Time (ms)	Time (%)
Qwen3-30B-A3B		
Overall generating prune plan (total)	10063.12	100%
– *Inter-layer coverage search*	87.43	0.87%
Smooth weights	29.36	0.29%
Compute prefix sum	19.91	0.20%
Binary search	37.15	0.37%
– *Intra-layer coverage search*	9619.28	95.59%
Recompute prefix sum	0.09	<0.01%
Binary Search (MoE 48 layers)	9619.19	95.59%
– per layer	200.40	1.99%
– *Alignment-aware redistribution*	356.41	3.54%
Compute K^{base}	20.44	0.20%
Compute headroom	17.02	0.17%
Allocate chunks	317.40	3.15%
Clamp to I	1.55	0.02%

Method2: Attribution-Guided Loss Approximation

Goal: **Accurately measure expert contribution while avoiding the high cost of ablating experts one by one.**

Key Idea: **Use a first-order approximation, currently optimized to a second-order approximation.**

Derivation

Step 1. *The output of one MoE block can be written as:*

$$y_\ell = \sum_{e \in \mathcal{E}_\ell} g_{\ell,e}(h_\ell)\, z_{\ell,e}$$

where $z_{l,e}$ is the expert output, and $g_{l,e}$ is the router weight assigned to that expert.

Step 2. Apply an expert-wise scaling factor β to the output:

$$\tilde{y}_{\ell,e}(h_\ell;\beta) = \beta \cdot y_{\ell,e}(h_\ell).$$

$$\beta = 0 : \text{expert removed}, \quad \beta = 1 : \text{expert retained}$$

Step 3. *Compute the loss against the original output*

$$\mathcal{L}(\beta) = \sum_{h_\ell \in \mathcal{D}} \left\| \hat{y}(h_\ell;\beta) - y^{\text{teacher}}(h_\ell) \right\|_2^2$$

where $\mathcal{D}$ is a small calibration set.

Method2: Attribution-Guided Loss Approximation

Goal: **Accurately measure expert contribution while avoiding the high cost of ablating experts one by one.**

Key Idea: **Use a first-order approximation, currently optimized to a second-order approximation.**

Derivation

Step 5*. Compute the first- and second-order derivatives at* $\beta = 1$

$$g_{\ell,e} = \frac{\partial \mathcal{L}(\beta)}{\partial \beta}\bigg|_{\beta=1}, \qquad h_{\ell,e} = \frac{\partial^2 \mathcal{L}(\beta)}{\partial \beta^2}\bigg|_{\beta=1}$$

Step 6. *Estimate expert contribution, i.e., sensitivity to expert removal.*

$$\Delta\mathcal{L}_{\ell,e} = \mathcal{L}(0) - \mathcal{L}(1) \approx -g_{\ell,e} + \frac{1}{2}h_{\ell,e} \qquad s_{\ell,e}^{expert} = \max\left(0,\ -g_{\ell,e} + \frac{1}{2}h_{\ell,e}\right)$$

Design Insight

- $\beta \in (0,1)$ simulates the output perturbation caused by pruning. $\Delta\mathcal{L}$ and the first-order gradient measures how sensitive the expert is to this perturbation.
- $\beta = 1$, both $\Delta\mathcal{L}$ and the first-order gradient are zero, The second-order term captures the curvature with respect to the scaling factor β, which reflects sensitivity to perturbation.

Method3: Alignment-Aware Redistribution

Goal: **Support mainstream quantization methods.**
Key Idea: **Regularize the post-pruning weight shapes.**

Motivation:
Without shape regularization, the deployed inference efficiency can be low.

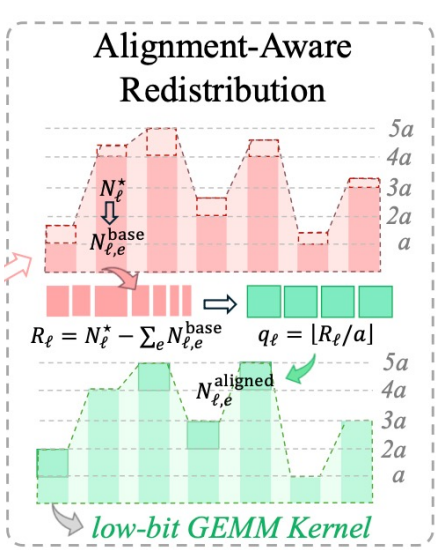
- If weights are explicitly padded to the kernel block size, such as 64, storage overhead is introduced. For example, on Qwen3-30B-A3B, this wastes 4.1% of weight storage.

- If explicit padding is not used and weights are only aligned to INT32, the computation falls back to a generic kernel and cannot benefit from low-bit GEMM acceleration.

Model	Alignment (a)	Tokens/s	Latency (ms)
Qwen1.5-MoE-A2.7B	–	31.41	31.84
	64	33.92	29.48
	128	37.12	26.94
Qwen3-30B-A3B	–	10.23	97.75
	64	12.98	77.03
	128	14.21	70.38

Method3: Alignment-Aware Redistribution

Goal: **Support mainstream quantization methods.**
Key Idea: **Regularize the post-pruning weight shapes.**



Algorithm Process: Based on Hamilton Apportionment

Step 1. *Round down to the minimum block size a supported by low-bit GEMM:*

$$N_{\ell,e}^{\text{base}} = \left\lfloor \frac{\tilde{N}_{\ell,e}}{a} \right\rfloor \cdot a$$

Step 2. *Compute the remaining allocation quota q_ℓ*

$$R_\ell = N_\ell^\star - \sum_{e\in[1,E]} N_{\ell,e}^{\text{base}}, \quad q_\ell = \left\lfloor \frac{R_\ell}{a} \right\rfloor$$

Step 3. *Compute the remaining number of channels, or the sum of remaining channel scores, for each expert:*

$$r_{\ell,e} = \frac{\tilde{N}_{\ell,e} - N_{\ell,e}^{\text{base}}}{a} \in [0,1)$$

Step 4. *Assign one additional block of a channels to the q_ℓ experts with the largest remaining channel count, or largest remaining score sum ($b_{l,e} = 1$):*

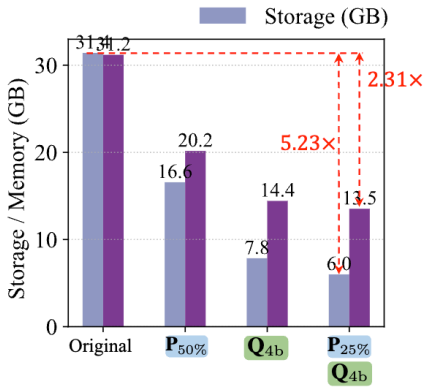
$$N'_{\ell,e} = N_{\ell,e}^{\text{base}} + a \cdot b_{\ell,e}$$

Results

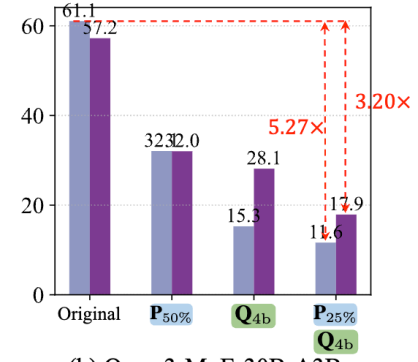
Ablation study

Strategy	ARC-c	GSM8K	HumanEval
Inter-layer sparsity allocation			
Non-prune	40.4	61.5	34.2
Uniform	37.3	57.8	30.5
U-shaped	37.2	57.6	29.3
Smoothed Loss	38.7	55.4	28.7
Coverage-based	–	–	–
↪ Uniform	38.0	57.0	29.3
↪ Raw loss	38.4	56.1	30.5
↪ **Smoothed Loss**	**40.0**	**58.2**	**30.5**
Intra-layer sparsity allocation			
Non-prune	40.4	61.5	34.2
Uniform	38.6	52.6	22.6
Loss-based	37.0	48.6	16.5
Router weight	38.2	57.3	22.6
Topk-Channel	37.7	50.6	29.9
Coverage-based	–	–	–
↪ Uniform	37.9	53.8	23.2
↪ Raw loss	36.1	45.7	12.2
↪ Router logits	37.5	55.2	23.8
↪ **Attribution**	**40.0**	**58.2**	**30.5**

Storage & Memory Reduction



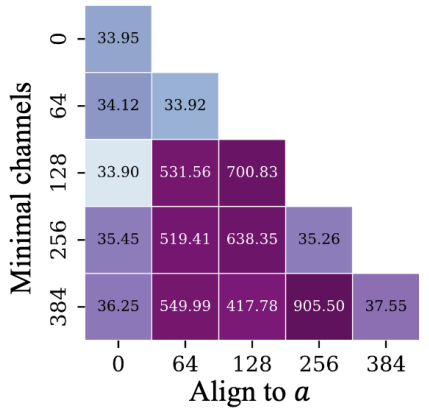
(a) DeepSeek-V2-Lite



(b) Qwen3-MoE-30B-A3B

Runtime Acceleration

(depending on whether the quantization library supports acceleration for the corresponding quantization operators)



(a) Throughput (Tokens/s)

Results

Accuracy Results on Commonsense QA Benchmarks

Method	Type	Storage	ARC-c	ARC-e	Hella	PiQA	BoolQ	Wino	MMLU	Avg
					Qwen1.5-MoE-A2.7B					
Baseline	–	28.67GB	40.41	69.44	77.17	80.79	79.57	69.77	61.08	68.32
Wanda	$P_{25\%}$	28.67GB	40.70	69.07	76.48	79.87	79.54	69.14	57.55	67.48
Wanda	$P_{50\%}$	28.67GB	38.48	65.99	71.92	78.07	76.57	66.61	53.82	64.49
EAC-MoE	$P_{38\%}$ $Q_{3.03b}$	4.35GB	41.30	66.71	74.97	79.16	75.35	68.75	55.96	61.77
MoE-I^2	$L_{53.98\%}$	15.18GB	41.13	71.68	53.08	–	75.08	66.54	–	57.82
PuzzleMoE	$M_{25\%}$	22.40GB	40.9	73.4	57.3	79.7	79.2	69.6	60.4	65.8
PuzzleMoE	$M_{50\%}$	16.17GB	40.7	73.5	56.5	79.4	78.6	69.4	60.0	65.4
Ours	$P_{50\%}$	**16.17GB**	**40.53**	**70.45**	**73.11**	**77.04**	**78.93**	**65.27**	**50.67**	**65.14**
Ours$_Q$	$P_{25\%}$ Q_{4b}	**5.60GB**	**45.13**	**73.19**	**75.01**	**79.00**	**84.22**	**68.67**	**58.29**	**69.07**
					Qwen3-MoE-30B-A3B					
Baseline	–	61.06GB	52.70	79.30	78.50	79.60	88.70	72.85	77.80	75.64
Wanda	$P_{25\%}$	61.06GB	26.45	33.71	25.43	52.12	38.50	51.14	22.95	35.76
Wanda	$P_{50\%}$	61.06GB	24.57	27.23	26.48	53.75	40.55	50.12	22.93	35.09
PuzzleMoE	$M_{25\%}$	46.56GB	51.6	78.9	58.3	79.3	88.2	70.4	76.6	71.9
PuzzleMoE	$M_{50\%}$	32.07GB	51.0	78.5	57.1	78.9	88.0	70.1	75.1	71.2
Ours	$P_{50\%}$	**32.07GB**	**52.13**	**78.03**	**77.19**	**79.33**	**83.36**	**71.59**	**61.27**	**70.81**
Ours$_Q$	$P_{25\%}$ Q_{4b}	**11.64GB**	**57.94**	**81.94**	**78.87**	**81.18**	**84.22**	**72.45**	**73.04**	**75.66**
					DeepSeek-V2-Lite					
Baseline	–	31.41GB	46.93	78.37	77.98	80.20	79.82	71.35	45.58	68.60
Wanda	$P_{25\%}$	31.41GB	46.84	76.64	58.44	79.65	79.39	71.51	53.84	66.62
Wanda	$P_{50\%}$	31.41GB	31.40	50.84	35.41	64.15	63.18	59.27	40.81	49.29
MoNE	$P_{25\%}$	24.00GB	46.67	74.62	43.00	79.76	78.47	71.43	–	65.65
MoNE	$P_{50\%}$	16.57GB	37.20	67.17	36.80	75.30	73.39	67.88	–	59.80
MoE-I^2	$L_{53.98\%}$	15.39GB	42.58	71.8	55.16	–	76.79	67.64	–	59.62
Ours	$P_{50\%}$	**16.57GB**	**42.49**	**73.78**	**74.59**	**78.40**	**71.77**	**69.22**	**44.16**	**64.92**
Ours$_Q$	$P_{25\%}$ Q_{4b}	**6.00GB**	**47.61**	**76.35**	**76.25**	**79.54**	**75.29**	**69.30**	**48.91**	**67.61**

Accuracy Results on Reasoning Tasks

Table 4. Reasoning benchmarks with math and code tasks.

Method	Type	GSM8K	HumanEval	MBPP
		Qwen1.5-MoE-A2.7B		
Baseline	–	61.5	34.2	36.6
C-Prune	$P_{20\%}$	39.4	32.9	–
Ours	$P_{50\%}$	**58.2**	**38.1**	**36.6**
Ours$_Q$	$P_{25\%}$ Q_{4b}	**58.5**	**36.6**	**38.5**
		DeepSeek-V2-Lite		
Baseline	–	30.9	32.3	43.2
C-Prune	$P_{20\%}$	26.4	18.9	–
Ours	$P_{50\%}$	**33.7**	**28.7**	**44.4**
Ours$_Q$	$P_{25\%}$ Q_{4b}	**29.3**	**23.8**	**37.4**
Method	**Type**	**MATH500**	**AIME25**	**LCB**
		Qwen3-MoE-30B-A3B		
Baseline	–	92.8	76.7	62.6
Ours	$P_{50\%}$	**95.0**	**64.0**	**–**
Ours$_Q$	$P_{25\%}$ Q_{4b}	**94.5**	**75.0**	**51.0**

Future Work

- Evaluate on more tasks
- Combine with other compression methods
- System-level optimization

ICML 2026 Spotlight

Thanks for listening!

GitHub

OpenReview.net

code link: https://github.com/yifu-ding/MoE-Slimming

paper link: https://openreview.net/pdf?id=oreET6Wz52