Modality-Aware Expert Pruning for MoE-Based Multimodal Large Language Models

Donggeun Kim Nota Inc. donggeun.kim@nota.ai Code: github.com/nota-github/MAEP

Idea in one line: score each expert separately for text and for vision, then keep what text needs and prune what only background image tokens use.

Motivation

Experts fill GPU memory, but only a few work per token

- Only 8 of 128 experts run for each token, yet all 128 must stay in GPU memory.
- Routed experts take **54 GB of the ~58 GB** BF16 weights in 30B-A3B models such as InternVL3.5 and Qwen3-VL.
- Existing expert pruning scores text and image tokens the same way, although an expert can matter for text but not for vision.

What pruning 50% of experts gives

InternVL3.5-30B-A3B on one A100, Hugging Face Transformers

57.5 GB → 30.5 GB

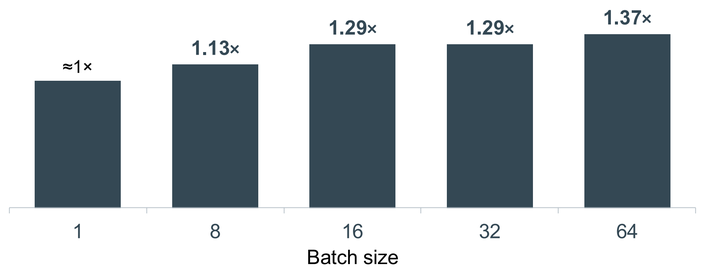
- **47% less weight memory:** the model now fits in a single 40 GB GPU.

1.52 → 5.88 tokens/s (3.9×)

- **Unpruned on a 40 GB GPU:** ~33% of the weights must be offloaded to CPU, so decoding is slow.
- **50% pruned:** everything stays on the GPU and decoding is 3.9× faster.

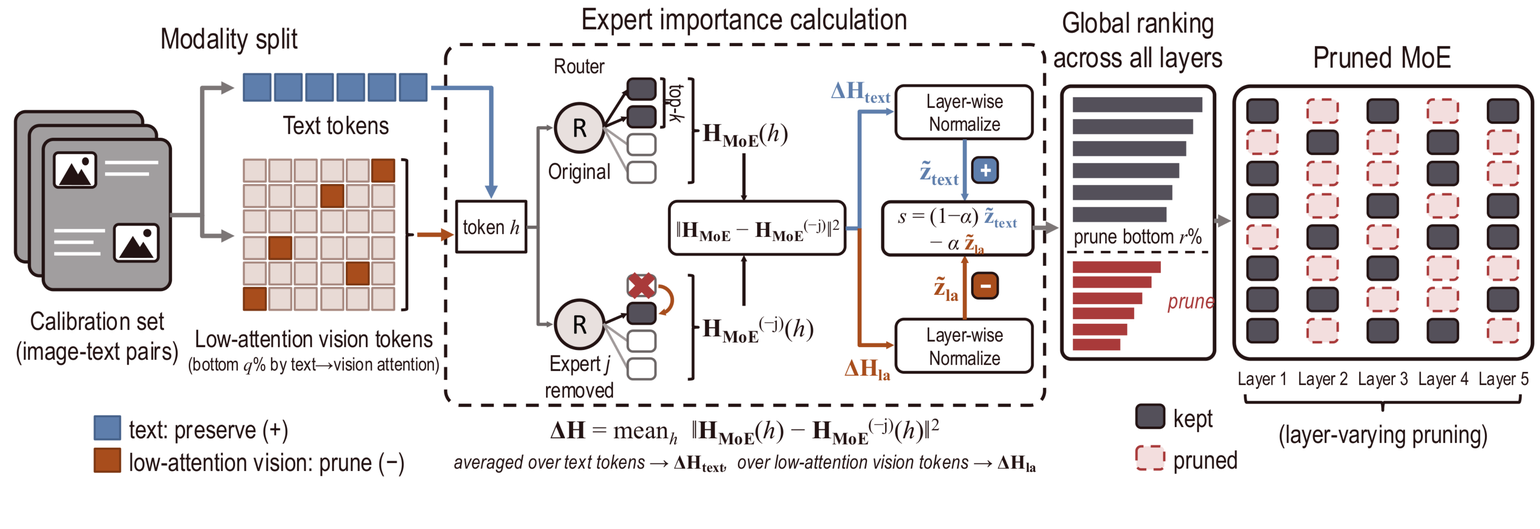
Bigger batch, bigger speed-up

Decoding speed of the pruned model ÷ unpruned model, both fully on the GPU



Why: each decoding step loads every expert the batch touches (about 55 per layer at batch 64, about 39 after pruning). Fewer experts means less memory traffic. At batch 1 the speed is unchanged because each token still uses 8 experts.

MAEP: how it works



1. Score every expert with ΔH

$$\Delta H_j = \text{mean}_h \, \| H_{MoE}(h) - H_{MoE}^{(-j)}(h) \|^2$$

$H_{MoE}(h)$: the MoE layer's output for token h. $H_{MoE}^{(-j)}(h)$: the same output with expert j removed and its routing weight passed to the remaining experts. A small ΔH means the others already cover expert j.

2. Split the score by token type

- **Text tokens:** high ΔH means the expert matters for language → keep.
- **Low-attention image tokens:** high ΔH means the expert mostly serves background → prune.

3. Combine and prune across all layers

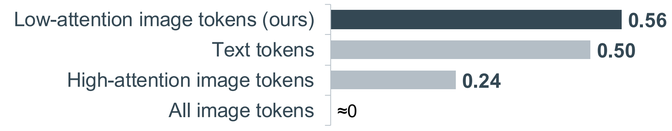
$$\text{score}_j = (1 - \alpha) \cdot \tilde{z}_{\text{text}} - \alpha \cdot \tilde{z}_{\text{low-attn}}$$

$\tilde{z}$ = ΔH z-scored per layer, α from calibration statistics. Prune the lowest scores globally, so each layer loses a different number.

Why split the score by modality?

- **Text and vision disagree on which experts matter:** the two rankings are unrelated (Spearman ρ ≈ −0.1), and removing an expert shifts the output up to 29× more at image positions than at text positions.

Why low-attention image tokens?



Spearman ρ between expert rankings from ShareGPT4V and ChartNet calibration (InternVL3.5-30B-A3B). Low-/high-attention = bottom 2% / top 20% of text→image attention.

- **Low-attention image tokens:** mostly background, so their expert ranking barely depends on the dataset.
- **All image tokens:** the ranking follows the image content and changes with the dataset.

Takeaways

Tested on InternVL3.5-30B-A3B, Qwen3-VL-30B-A3B and Gemma 4 26B-A4B (128 experts per layer, top-8 routing)

- **Idea:** text and vision use experts differently, so score each expert on text (what to keep) and on low-attention image tokens (what to prune).
- **Evidence:** expert rankings from low-attention image tokens stay stable across datasets (ρ ≈ 0.56); from all image tokens they do not (≈ 0).
- **Result:** 25% pruning keeps 97–99% (multimodal) / 92–95% (text-only) of the unpruned score; 50% keeps 89–94% / 78–84%.