M^{3}MoE: Multi-Level Routing Capacity Allocation for Extremely Sparse Multimodal MoE Inference

Anonymous authors
Paper under double-blind review

Abstract

Mixture-of-Experts (MoE) has become a prevalent architecture for scaling multimodal large language models (MLLMs), yet activating multiple experts for hundreds of visual tokens can still incur substantial inference overhead. In this work, we investigate routing redundancy in MoE-based MLLMs and identify two complementary forms of heterogeneity: (*i*) different MoE layers exhibit markedly different sensitivities to routing-capacity reduction, and (*ii*) visual tokens within the same layer require substantially different amounts of expert computation. Motivated by these observations, we propose M^{3}MoE, a multimodal-specific multi-level routing-capacity allocation framework for efficient MoE inference under a fixed visual-token expert budget. M^{3}MoE first profiles layer sensitivity through output-distribution perturbation and allocates routing capacity across MoE layers accordingly. It then redistributes each layer's budget among visual tokens by prioritizing token-expert pairs with larger routing weights, thereby preserving more routing mass under the same budget. Experiments on two representative MoE-based MLLMs across eight vision-language benchmarks show that M^{3}MoE achieves the highest normalized average performance compared with existing routing strategies across multiple capacity budgets. Under aggressive sparsity, it improves normalized average performance by up to 5.57% over the strongest baseline while skipping more than 91% of visual-token expert activations, and achieves up to 2.62× prefill speedup on Kimi-VL-A3B-Instruct.

1 Introduction

Multimodal Large Language Models (MLLMs) have recently achieved remarkable success across a wide range of vision-language tasks by integrating powerful large language models with visual encoders (Lu et al., 2024a; Lin et al., 2025; Wang et al., 2025; Team et al., 2026). To further scale model capacity while maintaining practical inference efficiency, recent MLLMs increasingly adopt Mixture-of-Experts (MoE) architectures, where only a small subset of experts is activated for each token. Representative models such as Kimi-VL (Team et al., 2025) and Qwen3-VL (Bai et al., 2025a) demonstrate that sparse expert routing provides an effective way to increase model capacity without proportionally increasing the computational cost.

Despite sparse activation, MoE-based MLLMs still incur substantial inference overhead (Huang et al., 2026; Xia et al., 2026). Compared with text-only LLMs, MLLMs typically process hundreds of visual tokens for each image, making expert computation in MoE layers a dominant bottleneck during inference. Existing work has explored multiple directions for efficient MoE inference, including expert skipping (Lu et al., 2024b; Huang et al., 2025b; Liu et al., 2026), pruning (Lee et al., 2025; Xu et al., 2026; Chen et al., 2025a), quantization (Duanmu et al., 2025; Chen et al., 2025b; Huang et al., 2025a), and system-level optimization (Kamahori et al., 2025; Yu et al., 2026). More recently, routing-capacity reduction has also been studied for multimodal MoE models. For example, FastMMoE (Xia et al., 2026) reduces expert computation through layer-wise routing adjustment, while MoDES (Huang et al., 2026) dynamically skips low-weight experts based on layer importance and routing weights. Methods developed for text-only MoE LLMs, such as Alloc-MoE (Liu et al., 2026), further demonstrate that routing capacity can be allocated adaptively across both layers and

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

tokens. However, how routing capacity should be allocated for visual tokens in multimodal MoE models remains insufficiently understood, particularly under aggressive capacity reduction.

In this work, we revisit this problem from the perspective of multimodal routing heterogeneity. Through systematic profiling, we observe that routing redundancy in MoE-based MLLMs exhibits two complementary structures. First, different MoE layers exhibit significantly different sensitivities to routing capacity reduction. As shown in Fig. 1 *Left*, reducing the routing capacity of some layers leads to a substantial shift in the output distribution, whereas others tolerate much more aggressive reduction. Second, visual tokens within the same layer exhibit highly diverse routing-weight distributions. As demonstrated in Fig. 1 *Right*, some tokens concentrate most of their routing mass on only a few experts, while others distribute it more broadly and therefore require more expert activations to preserve the same amount of routing mass. These observations suggest that allocating an identical routing capacity across layers or visual tokens can be inefficient, especially when the available expert budget becomes highly constrained.

Motivated by these observations, we propose **M^{3}MoE**, a multi-level routing capacity allocation framework for MoE-based MLLMs. M^{3}MoE operates at two complementary levels under a fixed visual-token expert budget. At the layer level, *Layer-level Sensitivity Profiling (LSP)* measures the effect of isolated routing-capacity reductions on the model's predictive distribution. The resulting sensitivity profiles guide *Sensitivity-Guided Layer-wise Allocation (SLA)*, which assigns more routing capacity to layers that are less tolerant to expert reduction. At the token level, *Token-wise Adaptive Routing (TAR)* redistributes each layer's allocated capacity among visual tokens according to their routing weights. By exploiting routing heterogeneity at both layer and token levels, M^{3}MoE enables more effective allocation of routing capacity under aggressive capacity reduction. Our contributions are summarized as follows:

- We systematically characterize routing heterogeneity in MoE-based MLLMs from both layer and token perspectives, revealing that both MoE layers and visual tokens exhibit substantially different sensitivities and computational requirements under sparse routing.
- We propose **M^{3}MoE**, a routing-capacity allocation framework for MoE-based MLLMs that uses perturbation-based layer sensitivity to allocate a fixed visual-token expert budget across layers, then redistributes each layer's budget through global token–expert ranking based on routing weights.
- Extensive experiments on representative MoE-based MLLMs demonstrate that M^{3}MoE consistently outperforms existing routing strategies across different sparsity levels, with increasingly larger advantages under more aggressive routing capacity reduction.

2 Related Work

2.1 MoE-based MLLMs

Recent advances in multimodal large language models (MLLMs) have significantly improved performance across a wide range of vision-language tasks. Representative dense models, such as DeepSeek-VL (Lu et al., 2024a) and Qwen2.5-VL (Bai et al., 2025b), achieve strong multimodal capabilities through large-scale multimodal pre-training and instruction tuning. However, continuously scaling dense architectures leads to rapidly increasing computational and memory cost during inference. To improve model scalability, recent MLLMs have increasingly adopted the Mixture-of-Experts (MoE) architecture. Representative models, including Kimi-VL (Team et al., 2025) and Qwen3-VL (Bai et al., 2025a), replace dense feed-forward networks with sparsely activated expert modules, allowing model capacity to grow without proportionally increasing per-token computation.

Despite sparse expert activation, MoE-based MLLMs can still incur considerable inference overhead due to hundreds of visual tokens. This characteristic makes routing efficiency particularly important for MoE-based MLLMs and motivates methods that reduce unnecessary expert activation while preserving multimodal capability.

2.2 Efficient MoE Inference

Efficient MoE inference has been studied from multiple perspectives. For routing-level acceleration, NAEE (Lu et al., 2024b) and AdapMoE (Zhong et al., 2025) dynamically skip low-weight experts during inference. Camera (Xu et al., 2026) and EAC-MoE (Chen et al., 2025a) exploit structural redundancy through expert pruning and quantization. Another closely related line of work explicitly considers routing-capacity allocation. Alloc-MoE (Liu et al., 2026) allocates expert-activation budgets across both layers and tokens for text-only MoE LLMs, demonstrating that routing capacity can benefit from adaptive allocation at multiple granularities. More recently, several methods have targeted the routing overhead of MoE-based MLLMs. FastMMoE (Xia et al., 2026) reduces visual-token expert computation using a depth-dependent layer-wise routing strategy. MoDES (Huang et al., 2026) combines global layer-wise importance with token-level expert importance to dynamically skip less important expert activations. These studies establish that both layer-wise and token-wise routing redundancy can be exploited for efficient multimodal MoE inference.

Different from these approaches, M^{3}MoE focuses on routing-capacity allocation for visual tokens under a fixed global expert budget. It estimates layer sensitivity through routing-capacity perturbation, allocates the global budget across layers accordingly, and further redistributes each layer's budget through global token–expert ranking. This formulation enables fine-grained routing-capacity allocation under aggressive sparsity.

3 Motivation

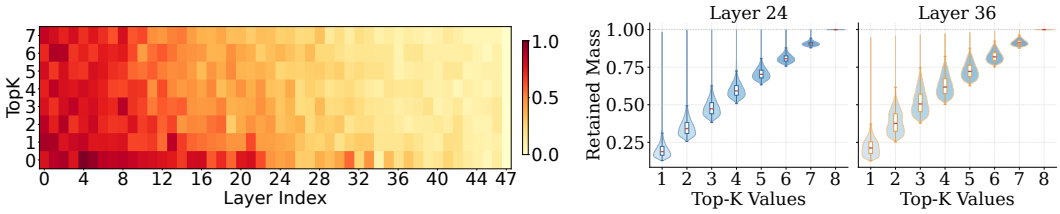

Figure 1: *Left*: Routing capacity reduction sensitivity of Qwen3-VL-30B-A3B-Instruct. Larger values indicate greater sensitivity. As detailed in Eq. equation 3. *Right*: Distribution of retained routing mass across visual tokens within two representative layers of Qwen3-VL-30B-A3B-Instruct. As defined in Eq. equation 6.

3.1 Layer-level Heterogeneity in MoE-based MLLMs

As shown in Fig. 1 *Left*, shallower layers exhibit substantially larger output distribution shifts (measured by JS divergence) under the same routing capacity, whereas deeper layers remain considerably more robust, indicating higher redundancy in their expert computations. Furthermore, the degradation trends differ markedly across layers as the routing capacity decreases, suggesting that the appropriate routing capacity is inherently layer-dependent rather than uniform. A similar layer-dependent sensitivity pattern is also observed on Kimi-VL-A3B-Instruct, as shown in Appendix A, indicating that this phenomenon is not specific to Qwen3-VL-30B-A3B-Instruct.

Motivation (i): The observation reveals substantial layer-wise heterogeneity in MoE-based MLLMs, motivating a layer-aware routing capacity allocation strategy that allocates the routing capacities according to layer sensitivity.

3.2 Token-level Heterogeneity in MoE-based MLLMs

Fig. 1 *Right* reveals substantial token-level heterogeneity in routing mass distributions. Although increasing Top-K value consistently improves the retained routing mass, the improvement quickly saturates, indicating diminishing returns from activating additional experts. More importantly, under the same Top-K value, the retained routing mass varies substantially across visual tokens. For example, with Top-1 routing, some visual tokens already retain more than 95% of the routing mass,

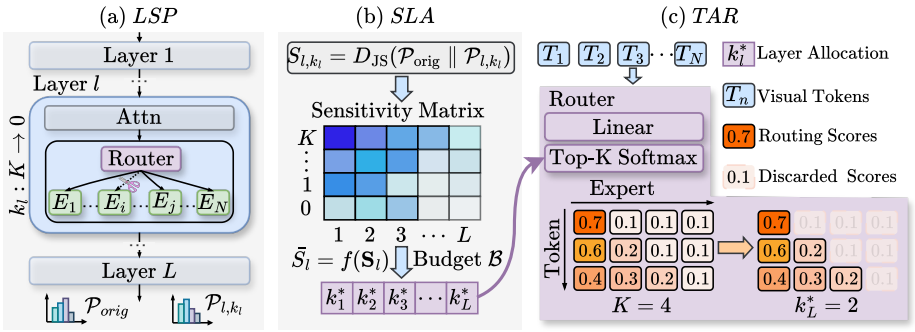

Figure 2: Overview of M^{3}MoE. (a) *Layer-level Sensitivity Profiling (LSP)*: Profile layer sensitivity by progressively reducing the target layer's routing capacity and measuring the JS divergence to the original output distributions. (b) *Sensitivity-Guided Layer-wise Allocation (SLA)*: Allocate layer-wise routing capacities according to the profiled sensitivities under a fixed global capacity budget. (c) *Token-wise Adaptive Routing (TAR)*: Redistribute each layer's routing capacity among visual tokens through global token-expert ranking.

whereas others retain less than 20%, suggesting highly diverse routing capacity requirements. We observe similar within-layer heterogeneity across additional layers; the corresponding results are provided in Appendix B.

Motivation (ii): The observation suggests that different visual tokens require different routing capacities to preserve sufficient routing mass, motivating a token-aware adaptive routing strategy within each layer.

4 Methodology

Based on the systematic analysis above, we propose M^{3}MoE, a multi-level routing capacity allocation framework for MoE-based MLLMs. As illustrated in Fig. 2, M^{3}MoE consists of three components: *Layer-level Sensitivity Profiling (LSP)* (Sec. 4.2), which profiles the sensitivity of each MoE layer to routing capacity reduction; *Sensitivity-Guided Layer-wise Allocation (SLA)* (Sec. 4.3), which allocates routing capacity across layers under a fixed capacity budget; and *Token-wise Adaptive Routing (TAR)* (Sec. 4.4), which redistributes each layer's routing capacity among visual tokens based on their routing weights.

4.1 Preliminaries

Mixture-of-Experts Routing. An MoE layer replaces the standard feed-forward network with a router and N experts $\mathcal{E} = \{E_1, E_2, \ldots, E_N\}$. Given an input token representation x, the router predicts routing weights over all experts:

$$w_i(x) = \frac{\exp(W_i x)}{\sum_{j=1}^{N} \exp(W_j x)}, \tag{1}$$

where W_i denotes the routing parameter associated with the i-th expert. The router then activates the Top-K experts with the largest routing weights, followed by weight renormalization over the selected experts. The output is computed as:

$$y = \sum_{i \in \mathcal{S}(x)} \hat{w}_i(x) E_i(x), \tag{2}$$

where $E_i(\cdot)$ denotes the i-th expert, $\mathcal{S}(x)$ denotes the set of selected Top-K experts and $\hat{w}_i(x)$ is the renormalized weight for expert i within this set.

4.2 LAYER-LEVEL SENSITIVITY PROFILING

Based on *Motivation (i)* in Sec. 3.1, we propose *Layer-level Sensitivity Profiling (LSP)* to quantify the sensitivity of each MoE layer to routing capacity reduction.

Consider an MoE-based MLLM with L MoE layers, where each layer activates $K_{\max}$ experts per token under the default routing strategy. We first perform a standard forward pass on the calibration set and denote the resulting predictive probability distribution of the first generated token as $\mathcal{P}_{\text{orig}}$. To profile the sensitivity of layer $l \in \{1, \dots, L\}$, we apply an isolated perturbation by reducing its Top-K from $K_{\max}$ to k_l ($k_l \in \{0, 1, \dots, K_{\max}-1\}$), while keeping all other MoE layers unchanged. The resulting predictive distribution is denoted by $\mathcal{P}_{l,k_l}$.

The sensitivity of layer l under Top-K setting k_l is quantified by the Jensen–Shannon (JS) divergence between the original and perturbed distributions:

$$S_{l,k_l} = D_{\text{JS}}(\mathcal{P}_{\text{orig}} \parallel \mathcal{P}_{l,k_l}), \tag{3}$$

where larger values indicate greater sensitivity to routing capacity reduction and therefore require greater routing capacity, whereas smaller values imply higher redundancy and can tolerate more aggressive routing capacity reduction. Since JS divergence is symmetric and bounded, it provides a stable and directly comparable metric to measure the sensitivity of routing distributions under different Top-K settings. Finally, we average the sensitivity scores over the calibration set to obtain the layer-wise sensitivity matrix $\mathbf{S} \in \mathbb{R}^{L \times K_{\max}}$.

4.3 SENSITIVITY-GUIDED LAYER-WISE ALLOCATION

As shown in Fig. 1 *Left*, different MoE layers tolerate routing-capacity reduction to substantially different degrees. Rather than assigning the same routing capacity to every layer or relying on a predefined depth-based schedule, we propose *Sensitivity-Guided Layer-wise Allocation (SLA)*. The intuition is that layers whose output distributions are more strongly affected by routing-capacity reduction should preserve more expert activations, whereas less sensitive layers can tolerate more aggressive reduction.

Formally, let $\mathcal{B}$ denote the total visual-token routing capacity budget across all L MoE layers. For each layer l, *LSP* provides a sensitivity profile $\mathbf{S}_l = \{\mathbf{S}_{l,k_l}\}_{k_l=0}^{K_{\max}-1}$, which characterizes the output-distribution perturbation induced by different routing capacities. To obtain a layer-level sensitivity signal that summarizes its behavior across different reduction levels, we aggregate the profile as

$$\bar{S}_l = \frac{1}{K_{\max}} \sum_{k=0}^{K_{\max}-1} S_{l,k}. \tag{4}$$

A larger $\bar{S}_l$ indicates that the model output is more sensitive to reducing the routing capacity of layer l, suggesting that this layer should retain a larger share of the available expert budget.

We convert the relative layer sensitivities into a bounded continuous allocation by proportionally distributing the global budget, capping any layer at $K_{\max}$, and redistributing the residual budget among unsaturated layers until

$$\sum_{l=1}^{L} \tilde{k}_l = \mathcal{B}, \qquad 0 \le \tilde{k}_l \le K_{\max}. \tag{5}$$

We then set $k_l^* = \lfloor \tilde{k}_l \rfloor$ and assign the remaining capacity to layers with the largest fractional parts until $\sum_{l=1}^{L} k_l^* = \mathcal{B}$.

Notably, the fine-grained sensitivity values $S_{l,k}$ are obtained through isolated single-layer perturbations, where all other layers remain at their default routing capacities. Directly optimizing the sum of these values across layers therefore implicitly treats independently profiled sensitivities as an additive objective under simultaneous multi-layer reduction. SLA instead uses the complete sensitivity profile to derive a compact layer-level allocation signal without making this additivity assumption. We empirically compare SLA with an exact dynamic-programming allocator that directly optimizes the full layer-capacity sensitivity matrix in Appendix G.

4.4 TOKEN-WISE ADAPTIVE ROUTING

While *SLA* determines how the global visual-token expert budget is distributed across MoE layers, it does not specify how each layer's budget should be assigned among individual visual tokens. Based on Motivation (*ii*), we propose *Token-wise Adaptive Routing (TAR)*, which redistributes each layer's expert budget among visual tokens according to their routing weights.

Consider the l-th MoE layer with routing capacity k_l^*. Let N_v denote the number of visual tokens in a sequence. We preserve the default capacity $K_{\max}$ for text tokens and apply routing-capacity reduction only to visual tokens. This design choice is empirically validated in Sec. 5.4. For visual tokens, the total routing capacity is: $\mathcal{B}_l = N_v \times k_l^*$. Let $\hat{w}_{t,j}$ denote the normalized routing weight of the j-th highest-ranked expert for visual token t, where normalization is performed over the original Top-$K_{\max}$ selected experts. For a token assigned with k experts, the retained routing mass is:

$$R_t(k) = \sum_{j=1}^{k} \hat{w}_{t,j}, \tag{6}$$

which measures the proportion of the routing mass preserved after routing capacity reduction. Let $\mathcal{A}$ denote the set of activated token–expert pairs, and let $\mathcal{A}_t = \{(t,j) \in \mathcal{A}\}$ denote the activated experts assigned to token t. *TAR* seeks to maximize the total retained routing mass under the capacity budget constraint:

$$\begin{aligned} \max_{\mathcal{A}} \quad & \sum_{(t,j)\in\mathcal{A}} \hat{w}_{t,j} \\ \text{s.t.} \quad & |\mathcal{A}| = \mathcal{B}_l, \quad 0 \le |\mathcal{A}_t| \le K_{\max}, \quad \forall t \in \mathcal{V}, \end{aligned} \tag{7}$$

where $|\mathcal{A}_t| = 0$ indicates that token t skips all routed experts. Since all routing weights are non-negative, the optimal solution is obtained by globally selecting the $\mathcal{B}_l$ largest routing weights across all token–expert pairs. Therefore, tokens with concentrated routing weights preserve most of their routing mass using fewer experts, whereas tokens with more distributed routing weights automatically receive additional expert activations. This adaptive allocation maximizes the total retained routing mass under the same capacity budget while introducing only one additional Top-K selection with negligible computational overhead.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Models and Benchmarks. We evaluate M^{3}MoE on Kimi-VL-A3B-Instruct (Team et al., 2025) and Qwen3-VL-30B-A3B-Instruct (Bai et al., 2025a). For brevity, we refer to them as Kimi-VL and Qwen3-VL in the subsequent experiments. As shown in Tab. 1, these two models differ in architecture, scale, MoE configurations and router characteristics. We evaluate our method on eight benchmarks, which can be grouped into three categories: OCR-intensive (TextVQA$_{\text{val}}$ (Singh et al., 2019), DocVQA$_{\text{val,lite}}$ (Mathew et al., 2021), OCRBench (Liu et al., 2024b)), diagram and chart reasoning (ChartQA (Masry et al., 2022), AI2D (Kembhavi et al., 2016)), and general-purpose multimodal understanding (MMStar (Chen et al., 2024), MMBench$_{\text{dev,en}}$ (Liu et al., 2024a), MME (Fu et al., 2025)), covering a broad spectrum from fine-grained textual perception to high-level visual reasoning.

Table 1: Comparison of LLM backbone architectures for the two evaluation models.

Model	# MoE Layers	# Act. / Tot. Params.	# Act. / Tot. Experts
Kimi-VL	26	2.4B / 15.7B	6 / 64
Qwen3-VL	48	3.3B / 30.5B	8 / 128

Baselines. We compare M^{3}MoE with several representative routing-capacity reduction methods. Specifically, **Uniform** assigns the same routing capacity to all MoE layers and visual tokens. We omit it at $\bar{K} = 0.5$, since a uniform per-token routing capacity below one expert cannot be implemented with standard integer Top-K routing. **Ascend** and **Descend** serve as heuristic layer-wise allocation baselines, distributing routing capacity monotonically across layers in opposite directions. We adapt **Alloc-MoE** (Liu et al., 2026) to visual tokens in MoE-based MLLMs. For MLLM-specific

baselines, we compare with **FastMMoE** (Xia et al., 2026) and **MoDES** (Huang et al., 2026). Detailed adaptations, hyperparameter settings, and budget-matching procedures are provided in Appendix C.

Implementation Details. We follow the official `lmms-eval` (Zhang et al., 2025) benchmark for performance evaluation. For layer-wise sensitivity profiling, we randomly sample 128 examples from GQA (Hudson & Manning, 2019) as the calibration set. Ablation studies on the calibration set are presented in Sec. 5.4. Since visual tokens are processed only during the prefill stage, we report prefill latency and speedup on Kimi-VL for inference efficiency evaluation. Details of our optimized prefill path are in Appendix E. All experiments are conducted on a single H100 80GB GPU.

5.2 Main Results

Table 2: Kimi-VL performance under different Average-K (average routing capacity of visual tokens across all MoE layers). Avg.(%) is the average normalized score across all benchmarks, each normalized by default $K = 6$. Best and second-best are bold and underlined, respectively. Parentheses report M^{3}MoE's improvement over the strongest baseline under the same budget.

Method	TextVQA	ChartQA	MMStar	MMBench	MME	AI2D	DocVQA	OCRBench	Avg.(%) ↑
K=6 (Default)	88.70	89.44	49.55	83.33	2201	83.61	97.78	83.60	100.00
$\bar{K} = 2$ *(Skip 66.67% Visual Expert Activations)*									
Uniform	87.85	88.28	**49.30**	82.65	2181	83.00	**97.62**	82.70	99.19
Ascend	80.05	70.48	46.48	77.41	1792	77.20	87.00	72.00	88.08
Descend	86.71	88.72	**49.30**	82.89	**2210**	83.10	97.30	82.20	99.19
FastMMoE	84.12	83.48	48.64	82.39	2198	**83.68**	97.16	79.60	97.47
Alloc-MoE	85.10	85.52	48.27	**83.08**	2179	83.48	97.22	80.40	97.89
MoDES	84.85	87.92	47.98	82.53	2173	82.68	97.02	79.40	97.71
M^{3}MoE (Ours)	**88.18**	**89.40**	49.15	82.90	2195	83.26	97.32	**84.10**	**99.69** (+0.50)
$\bar{K} = 1$ *(Skip 83.33% Visual Expert Activations)*									
Uniform	83.16	84.68	47.12	81.69	2101	81.90	94.17	74.50	95.05
Ascend	73.42	56.88	45.22	77.32	1724	76.88	81.67	63.40	82.51
Descend	81.71	79.88	47.44	80.76	2065	82.42	96.09	74.80	94.28
FastMMoE	80.27	73.88	46.47	79.12	1982	79.95	93.94	72.90	91.35
Alloc-MoE	80.89	77.16	47.38	79.38	1976	80.63	93.87	72.60	92.18
MoDES	81.84	84.80	47.38	81.75	2141	82.12	95.54	74.30	95.36
M^{3}MoE (Ours)	**86.94**	**88.40**	**48.67**	**82.22**	**2163**	**82.45**	**97.26**	**81.10**	**98.39** (+3.03)
$\bar{K} = 0.5$ *(Skip 91.67% Visual Expert Activations)*									
Ascend	73.36	56.24	45.27	77.06	1755	77.07	81.28	62.70	82.43
Descend	80.11	76.24	46.20	79.12	1986	80.89	93.81	72.00	91.60
FastMMoE	76.59	64.80	45.62	77.58	1794	78.40	87.51	66.80	86.08
Alloc-MoE	76.55	66.24	45.80	77.75	1810	78.34	87.74	67.70	86.59
MoDES	79.83	76.20	45.80	79.18	1871	80.34	91.49	70.40	90.19
M^{3}MoE (Ours)	**84.23**	**86.65**	**47.95**	**82.39**	**2156**	**82.80**	**97.06**	**78.30**	**97.17** (+5.57)

Overall Performance. Tab. 2 compares M^{3}MoE with representative routing baselines under different global capacity budgets on Kimi-VL. Across all Average-K settings, M^{3}MoE consistently achieves the highest average normalized performance, surpassing the strongest baseline by up to +5.57% while retaining 97.17% of the default performance under the most aggressive setting. Compared with heuristic layer-wise strategies, M^{3}MoE achieves substantially higher performance, demonstrating the importance of sensitivity-guided capacity allocation. It also consistently outperforms Alloc-MoE, suggesting the benefit of explicitly accounting for multimodal-specific routing heterogeneity when allocating visual-token capacity. Compared with MLLM-specific FastMMoE and MoDES, M^{3}MoE further demonstrates the effectiveness of combining sensitivity-guided layer allocation with routing-weight-based visual-token redistribution under the same routing budget.

Performance under Aggressive Routing Capacity Reduction. The advantage of M^{3}MoE becomes increasingly pronounced as the capacity budget decreases. On Kimi-VL, the improvement over the strongest baseline increases from 0.50% at $\bar{K} = 2$ to 3.03% at $\bar{K} = 1$, and further to

5.57% at $\bar{K} = 0.5$, indicating that M^{3}MoE utilizes constrained routing budgets more effectively, particularly under aggressive sparsity.

Generalization Across Models and Tasks. We further evaluate M^{3}MoE on Qwen3-VL, with complete results reported in Appendix D (Tab. 7). Despite architectural differences, M^{3}MoE shows consistent behavior across models and budgets, outperforming the strongest baseline on Qwen3-VL by 1.57%, 2.41%, and 3.01% at $\bar{K} = 2$, 1, and 0.5, respectively, while retaining 96.27% of the default performance at $\bar{K} = 0.5$. These results demonstrate that the proposed multi-level routing capacity allocation generalizes across different MoE-based MLLMs and downstream tasks.

5.3 Efficiency Analysis

Profiling Efficiency. Fig. 3 *(Left)* reports offline *LSP* latency on Kimi-VL and Qwen3-VL across calibration set sizes. It scales roughly linearly with sample count: 64 samples take 6 min for Kimi-VL and 27 min for Qwen3-VL. Profiling runs once per model; the sensitivity matrix is reused by lightweight *SLA* to generate layer-wise routing allocations for arbitrary global capacity budgets. Thus, profiling is fully amortized offline and adds no inference overhead.

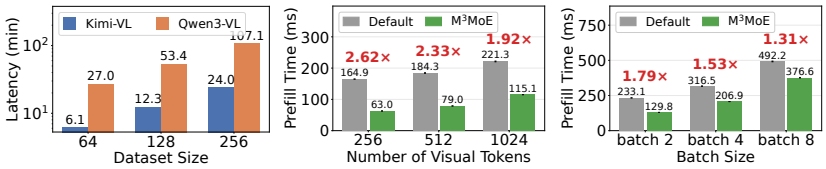

Figure 3: *(Left)*: sensitivity-analysis latency (min) on Kimi-VL and Qwen3-VL across different dataset sizes (batch size = 32). *(Middle & Right)*: prefill latency (ms) of Kimi-VL under *(Middle)* different numbers of visual tokens (batch size = 1) and *(Right)* different batch size (number of visual tokens = 512). Both are conducted under $\bar{K} = 0.5$.

Prefill Efficiency. We measure the prefill latency with different numbers of visual tokens while fixing the batch size to 1. As shown in Fig. 3 *(Middle & Right)*, our method consistently outperforms the original model, yielding 2.62×, 2.33×, and 1.92× speedup at 256, 512, and 1024 visual tokens, respectively. We further evaluate the cross-batch setting at 512 visual tokens: as the batch size increases to 2, 4, and 8, our method still achieves 1.80×, 1.53×, and 1.31× speedup, respectively. The results demonstrate that our proposed strategy maintains consistent latency improvements across visual sequence lengths and batch sizes, consistently translating the reduced MoE computation into prefill latency gains.

5.4 Ablation Studies

Table 3: Ablation of *SLA* and *TAR* on Kimi-VL. Starting from Descend, we evaluate +*SLA*, +*TAR*, and both (M3MoE).

Method	$\bar{K} = 2$	$\bar{K} = 1$	$\bar{K} = 0.5$
Descend	99.19	94.28	91.60
+ *SLA*	99.35 (+0.16)	97.57 (+3.29)	95.76 (+4.16)
+ *TAR*	99.44 (+0.25)	94.46 (+0.18)	92.05 (+0.45)
M^{3}MoE	**99.69 (+0.50)**	**98.39 (+4.11)**	**97.17 (+5.57)**

Table 4: *TAR* selection strategy ablation on Kimi-VL under the same *SLA* allocation. Parentheses denote performance change relative to *SLA*.

Strategy	$\bar{K} = 2$	$\bar{K} = 1$	$\bar{K} = 0.5$
SLA	99.35	97.57	95.76
+Random	97.22 (-2.13)	92.60 (-4.97)	89.75 (-6.01)
+Min	86.17 (-13.18)	83.75 (-13.82)	83.07 (-12.69)
+Top (Ours)	**99.69 (+0.34)**	**98.39 (+0.82)**	**97.17 (+1.41)**

Effect of Each Component. Tab. 3 shows that both SLA and TAR consistently improve upon the Descend baseline across all Average-K settings. Under extreme sparsity ($\bar{K} = 0.5$), SLA contributes the largest gain (+4.16%), highlighting the importance of effective layer-wise capacity allocation when the routing budget is highly constrained. TAR further refines the allocation within each layer, with its benefit naturally depending on the quality of the layer-level capacities provided by SLA. Combining both components consistently achieves the best performance, yielding up to a 5.57% improvement over Descend. These results demonstrate the complementary roles of SLA for layer-level allocation and TAR for token-level refinement.

Visual vs. Text Routing Reduction. To determine which token type is more suitable for routing reduction, we reduce visual and text routing from $K = 6$ to $K = 1$ separately. As shown in Tab. 5, visual-only reduction retains 95.05% of the default performance, compared with 90.17% for text-only reduction and 86.93% when both are reduced. This indicates that text-token routing is substantially more sensitive to aggressive capacity reduction. Since multimodal inputs also contain many more visual tokens, reducing visual-token routing provides a more favorable performance–efficiency trade-off. Accordingly, M^{3}MoE applies capacity reduction only to visual tokens.

Table 5: Effect of reducing routing capacity for different token types on Kimi-VL. Details are provided in Appendix F.

Routing Reduction	Avg.(%) ↑
None ($K = 6$)	100.00
Visual only ($K_v = 1$)	**95.05**
Text only ($K_t = 1$)	90.17
Visual + Text ($K_v = K_t = 1$)	86.93

Selection Strategy of *TAR*. To isolate the effect of token–expert selection in *TAR*, we compare the proposed Top strategy with Random selection and Min selection, which chooses the lowest-scoring pairs, while keeping the layer-wise capacities fixed. As shown in Tab. 4, Top consistently performs best, whereas Random and Min cause substantial degradation. The widening gap at lower budgets further indicates that effective token-wise selection becomes increasingly important under sparse expert activation, validating the proposed Top strategy.

Impact of Calibration Set and Sample Size. Tab. 6 evaluates how the type and size of the calibration set affect *LSP*. To isolate the effect of calibration, we evaluate the resulting *SLA* allocations without applying *TAR*. Overall, the layer-wise allocation is robust to both the calibration dataset and its size. Using either GQA (Hudson & Manning, 2019) or ScienceQA (Lu et al., 2022) yields similar performance across different sample sizes, and increasing the calibration set beyond 64 samples provides little additional benefit. These results indicate that stable layer sensitivity profiles and routing-capacity allocations can be obtained from a small calibration set.

Table 6: Ablation on calibration set type and size for *LSP* on Kimi-VL. Average performance (%) over $\bar{K} = 2, 1, 0.5$, normalized by default $K = 6$.

Dataset	32	64	128	256
GQA	97.75	97.70	97.56	97.56
ScienceQA	97.83	97.85	97.64	97.65

5.5 Visualization Analysis

Depth-dependent allocation. Fig. 4 shows that shallow layers retain more expert activations, while deep layers undergo greater reduction, with the final several layers skipping all routed experts for visual tokens. This trend agrees with the sensitivity analysis in Sec. 3.1 and the allocation pattern reported by AllocMoE (Liu et al., 2026), extending the observation to visual-token routing in multimodal MoE models.

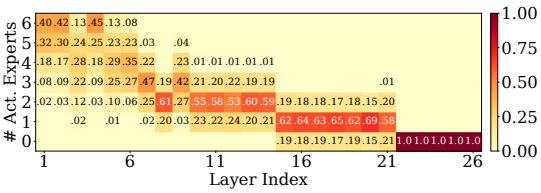

Figure 4: Distribution of visual-token routing capacities across MoE layers in Kimi-VL at $\bar{K} = 2$. Each cell indicates the fraction of visual tokens assigned the corresponding number of routed experts.

Within-layer heterogeneity. Visual tokens within the same layer receive different numbers of experts through TAR under the fixed layer budget. This complements the ablation results in Sec. 5.4, illustrating how SLA and TAR allocate capacity across layers and tokens, respectively.

6 Conclusion

In this paper, we propose M^{3}MoE, a multi-level routing capacity allocation framework for MoE-based MLLMs that exploits routing heterogeneity at both layer and token levels to optimize routing capacity allocation for visual tokens under a fixed capacity budget. Extensive experiments demonstrate that M^{3}MoE consistently outperforms existing representative routing strategies, with increasingly larger improvements under aggressive routing capacity reduction.

AI USE STATEMENT

Generative AI tools were used to assist with language polishing, grammatical correction, and consistency checking during the preparation of this manuscript. They were not used to generate experimental results or independently determine the scientific conclusions of this work. The authors reviewed and verified all AI-assisted content and take full responsibility for the accuracy, originality, and integrity of the manuscript.

REPRODUCIBILITY STATEMENT

The proposed method, experimental setup, baselines, and evaluation protocol are described in the main text and appendices. We provide implementation details for sensitivity profiling, routing-capacity allocation, and efficient prefill in the appendix. All datasets and benchmarks used in this work are publicly available.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report, 2025a. URL https://arxiv.org/abs/2511.21631.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025b. URL https://arxiv.org/abs/2502.13923.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. Are we on the right way for evaluating large vision-language models? In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 27056–27087. Curran Associates, Inc., 2024. doi: 10.52202/079017-0850. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/2f8ee6a3d766b426d2618e555b5aeb39-Paper-Conference.pdf.

Yuanteng Chen, Yuantian Shao, Peisong Wang, and Jian Cheng. EAC-MoE: Expert-selection aware compressor for mixture-of-experts large language models. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12942–12963, Vienna, Austria, July 2025a. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.633. URL https://aclanthology.org/2025.acl-long.633/.

Zhixuan Chen, Xing Hu, Dawei Yang, Zukang Xu, Xu Chen, Zhihang Yuan, Sifan Zhou, and Jiangyong Yu. MoEQuant: Enhancing quantization for mixture-of-experts large language models via expert-balanced sampling and affinity guidance. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 8245–8260. PMLR, 13–19 Jul 2025b. URL https://proceedings.mlr.press/v267/chen25aa.html.

Haojie Duanmu, Xiuhong Li, Zhihang Yuan, Size Zheng, Jiangfei Duan, Xingcheng Zhang, and Dahua Lin. MxMoE: Mixed-precision quantization for MoE with accuracy and performance co-design. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 14793–14806. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/duanmu25a.html.

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, Yunsheng Wu, Rongrong Ji, Caifeng Shan, and Ran He. Mme: A comprehensive evaluation benchmark for multimodal large language models. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38. Curran Associates, Inc., 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/d79a27cf2772fe00be7f341efc0eb517-Paper-Datasets_and_Benchmarks_Track.pdf.

Beichen Huang, Yueming Yuan, ZELEI SHAO, and Minjia Zhang. Milo: Efficient quantized moe inference with mixture of low-rank compensators. In M. Zaharia, G. Joshi, and Y. Lin (eds.), *Proceedings of Machine Learning and Systems*, volume 7. MLSys, 2025a. URL https://proceedings.mlsys.org/paper_files/paper/2025/file/9032e5c9ec394ce768a2fa9bdc56af6c-Paper-Conference.pdf.

Wei Huang, Yue Liao, Jianhui Liu, Ruifei He, Haoru Tan, Shiming Zhang, Hongsheng Li, Si Liu, and XIAOJUAN QI. Mixture compressor for mixture-of-experts llms gains more. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 68628–68648, 2025b. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/abc1943857a42935ceacff03c524bb44-Paper-Conference.pdf.

Yushi Huang, Zining Wang, Zhihang Yuan, Yifu Ding, Ruihao Gong, Jinyang Guo, Xianglong Liu, and Jun Zhang. Modes: Accelerating mixture-of-experts multimodal large language models via dynamic expert skipping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 30205–30215, June 2026.

Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6700–6709, 2019.

Keisuke Kamahori, Tian Tang, Yile Gu, Kan Zhu, and Baris Kasikci. Fiddler: Cpu-gpu orchestration for fast inference of mixture-of-experts models. In *International Conference on Learning Representations*, volume 2025, pp. 56099–56115, 2025.

Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A diagram is worth a dozen images. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (eds.), *Computer Vision – ECCV 2016*, pp. 235–251, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46493-0.

Jaeseong Lee, Seung-won Hwang, Aurick Qiao, Daniel F Campos, Zhewei Yao, and Yuxiong He. STUN: Structured-then-unstructured pruning for scalable MoE pruning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13660–13676, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.671. URL https://aclanthology.org/2025.acl-long.671/.

Zhiyu Lin, Yifei Gao, Xian Zhao, Yunfan Yang, and Jitao Sang. Mind with eyes: from language reasoning to multimodal reasoning, 2025. URL https://arxiv.org/abs/2503.18071.

Baihui Liu, Kaiyuan Tian, Wei Wang, Zhaoning Zhang, Linbo Qiao, and Dongsheng Li. AllocMoE: Budget-aware expert activation allocation for efficient mixture-of-experts inference. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the*

64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 9653–9667, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.437. URL https://aclanthology.org/2026.acl-long.437/.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. Mmbench: Is your multi-modal model an all-around player? In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 216–233, Cham, 2024a. Springer Nature Switzerland. ISBN 978-3-031-72658-3.

Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xu-Cheng Yin, Cheng-Lin Liu, Lianwen Jin, and Xiang Bai. Ocrbench: on the hidden mystery of ocr in large multimodal models. *Science China Information Sciences*, 67(12), December 2024b. ISSN 1869-1919. doi: 10.1007/s11432-024-4235-6. URL http://dx.doi.org/10.1007/s11432-024-4235-6.

Haoyu Lu, Wen Liu, Bo Zhang, Bingxuan Wang, Kai Dong, Bo Liu, Jingxiang Sun, Tongzheng Ren, Zhuoshu Li, Hao Yang, Yaofeng Sun, Chengqi Deng, Hanwei Xu, Zhenda Xie, and Chong Ruan. Deepseek-vl: Towards real-world vision-language understanding, 2024a. URL https://arxiv.org/abs/2403.05525.

Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *The 36th Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

Xudong Lu, Qi Liu, Yuhui Xu, Aojun Zhou, Siyuan Huang, Bo Zhang, Junchi Yan, and Hongsheng Li. Not all experts are equal: Efficient expert pruning and skipping for mixture-of-experts large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6159–6172, Bangkok, Thailand, August 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.334. URL https://aclanthology.org/2024.acl-long.334/.

Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. ChartQA: A benchmark for question answering about charts with visual and logical reasoning. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 2263–2279, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.177. URL https://aclanthology.org/2022.findings-acl.177/.

Minesh Mathew, Dimosthenis Karatzas, and C.V. Jawahar. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 2200–2209, January 2021.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

Kimi Team, Angang Du, Bohong Yin, Bowei Xing, Bowen Qu, Bowen Wang, Cheng Chen, Chenlin Zhang, Chenzhuang Du, Chu Wei, Congcong Wang, Dehao Zhang, Dikang Du, Dongliang Wang, Enming Yuan, Enzhe Lu, Fang Li, Flood Sung, Guangda Wei, Guokun Lai, Han Zhu, Hao Ding, Hao Hu, Hao Yang, Hao Zhang, Haoning Wu, Haotian Yao, Haoyu Lu, Heng Wang, Hongcheng Gao, Huabin Zheng, Jiaming Li, Jianlin Su, Jianzhou Wang, Jiaqi Deng, Jiezhong Qiu, Jin Xie, Jinhong Wang, Jingyuan Liu, Junjie Yan, Kun Ouyang, Liang Chen, Lin Sui, Longhui Yu, Mengfan Dong, Mengnan Dong, Nuo Xu, Pengyu Cheng, Qizheng Gu, Runjie Zhou, Shaowei Liu, Sihan Cao, Tao Yu, Tianhui Song, Tongtong Bai, Wei Song, Weiran He, Weixiao Huang, Weixin Xu, Xiaokun Yuan, Xingcheng Yao, Xingzhe Wu, Xinxing Zu, Xinyu Zhou, Xinyuan Wang, Y. Charles, Yan Zhong, Yang Li, Yangyang Hu, Yanru Chen, Yejie Wang, Yibo Liu, Yibo Miao, Yidao Qin, Yimin Chen, Yiping Bao, Yiqin Wang, Yongsheng Kang, Yuanxin Liu, Yulun Du, Yuxin Wu, Yuzhi Wang, Yuzi Yan, Zaida Zhou, Zhaowei Li, Zhejun Jiang, Zheng Zhang, Zhilin

Yang, Zhiqi Huang, Zihao Huang, Zijia Zhao, and Ziwei Chen. Kimi-VL technical report, 2025. URL https://arxiv.org/abs/2504.07491.

V Team, Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, Shuaiqi Duan, Weihan Wang, Yan Wang, Yean Cheng, Zehai He, Zhe Su, Zhen Yang, Ziyang Pan, Aohan Zeng, Baoxu Wang, Bin Chen, Boyan Shi, Changyu Pang, Chenhui Zhang, Da Yin, Fan Yang, Guoqing Chen, Haochen Li, Jiale Zhu, Jiali Chen, Jiaxing Xu, Jiazheng Xu, Jing Chen, Jinghao Lin, Jinhao Chen, Jinjiang Wang, Junjie Chen, Leqi Lei, Letian Gong, Leyi Pan, Mingdao Liu, Mingde Xu, Mingzhi Zhang, Qinkai Zheng, Ruiliang Lyu, Shangqin Tu, Sheng Yang, Shengbiao Meng, Shi Zhong, Shiyu Huang, Shuyuan Zhao, Siyan Xue, Tianshu Zhang, Tianwei Luo, Tianxiang Hao, Tianyu Tong, Wei Jia, Wenkai Li, Xiao Liu, Xiaohan Zhang, Xin Lyu, Xinyu Zhang, Xinyue Fan, Xuancheng Huang, Yadong Xue, Yanfeng Wang, Yanling Wang, Yanzi Wang, Yifan An, Yifan Du, Yiheng Huang, Yilin Niu, Yiming Shi, Yu Wang, Yuan Wang, Yuanchang Yue, Yuchen Li, Yusen Liu, Yutao Zhang, Yuting Wang, Yuxuan Zhang, Zhao Xue, Zhengxiao Du, Zhenyu Hou, Zihan Wang, Peng Zhang, Debing Liu, Bin Xu, Juanzi Li, Minlie Huang, Yuxiao Dong, and Jie Tang. Glm-4.5v and glm-4.1v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning, 2026. URL https://arxiv.org/abs/2507.01006.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, Zhaokai Wang, Zhe Chen, Hongjie Zhang, Ganlin Yang, Haomin Wang, Qi Wei, Jinhui Yin, Wenhao Li, Erfei Cui, Guanzhou Chen, Zichen Ding, Changyao Tian, Zhenyu Wu, Jingjing Xie, Zehao Li, Bowen Yang, Yuchen Duan, Xuehui Wang, Zhi Hou, Haoran Hao, Tianyi Zhang, Songze Li, Xiangyu Zhao, Haodong Duan, Nianchen Deng, Bin Fu, Yinan He, Yi Wang, Conghui He, Botian Shi, Junjun He, Yingtong Xiong, Han Lv, Lijun Wu, Wenqi Shao, Kaipeng Zhang, Huipeng Deng, Biqing Qi, Jiaye Ge, Qipeng Guo, Wenwei Zhang, Songyang Zhang, Maosong Cao, Junyao Lin, Kexian Tang, Jianfei Gao, Haian Huang, Yuzhe Gu, Chengqi Lyu, Huanze Tang, Rui Wang, Haijun Lv, Wanli Ouyang, Limin Wang, Min Dou, Xizhou Zhu, Tong Lu, Dahua Lin, Jifeng Dai, Weijie Su, Bowen Zhou, Kai Chen, Yu Qiao, Wenhai Wang, and Gen Luo. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency, 2025. URL https://arxiv.org/abs/2508.18265.

Guoyang Xia, Yifeng Ding, Fengfa Li, Lei Ren, Wei Chen, Fangxiang Feng, and Xiaojie Wang. Fastmmoe: Accelerating multimodal large language models through dynamic expert activation and routing-aware token pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Findings*, pp. 5915–5924, June 2026.

Yuzhuang Xu, Xu Han, Yuanchi Zhang, Yixuan Wang, Yijun Liu, Shiyu Ji, Qingfu Zhu, and Wanxiang Che. Camera: Multi-matrix joint compression for moe models via micro-expert redundancy analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 27395–27404, 2026.

Enda Yu, Dezun Dong, Zhaoning Zhang, Zhe Bai, Weiling Yang, Haojie Wang, Dongsheng Li, Yongwei Wu, and Xiangke Liao. Layerscope: Predictive cross-layer scheduling for efficient multi-batch moe inference on legacy servers. In *Proceedings of the 40th ACM International Conference on Supercomputing*, pp. 881–892, 2026.

Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, and Ziwei Liu. LMMs-eval: Reality check on the evaluation of large multimodal models. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 881–916, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-7. doi: 10.18653/v1/2025.findings-naacl.51. URL https://aclanthology.org/2025.findings-naacl.51/.

Shuzhang Zhong, Ling Liang, Yuan Wang, Runsheng Wang, Ru Huang, and Meng Li. Adapmoe: Adaptive sensitivity-based expert gating and management for efficient moe inference. In *Proceedings of the 43rd IEEE/ACM International Conference on Computer-Aided Design*, ICCAD '24. Association for Computing Machinery, 2025. ISBN 9798400710773. doi: 10.1145/3676536.3676741. URL https://doi.org/10.1145/3676536.3676741.

A Layer-level Heterogeneity on Kimi-VL-A3B-Instruct

To examine whether the layer-level routing heterogeneity observed on Qwen3-VL-30B-A3B-Instruct is specific to a particular model architecture, we further perform the same sensitivity profiling analysis on Kimi-VL-A3B-Instruct. As shown in Fig. 5, Kimi-VL-A3B-Instruct exhibits a qualitatively similar layer-dependent sensitivity pattern. Earlier MoE layers are generally more sensitive to routing capacity reduction, whereas many later layers tolerate substantially more aggressive reduction with relatively small changes in the output distribution.

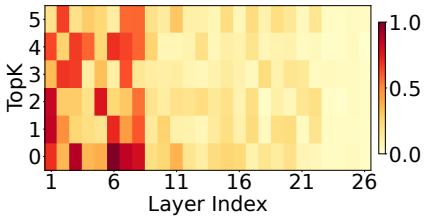

Figure 5: Routing capacity reduction sensitivity of Kimi-VL-A3B-Instruct. Larger values indicate greater sensitivity. As detailed in Eq. equation 3.

Importantly, the sensitivity pattern is not determined solely by layer depth: different layers still exhibit distinct responses as the routing capacity decreases. This observation further supports the use of model-specific sensitivity profiling rather than a fixed depth-based routing schedule. Together with the Qwen3-VL results in Fig. 1 *Left*, the analysis indicates that pronounced layer-level routing heterogeneity is present across the two evaluated MoE-based MLLMs.

B Additional Token-level Heterogeneity Analysis on Qwen3-VL-30B-A3B-Instruct

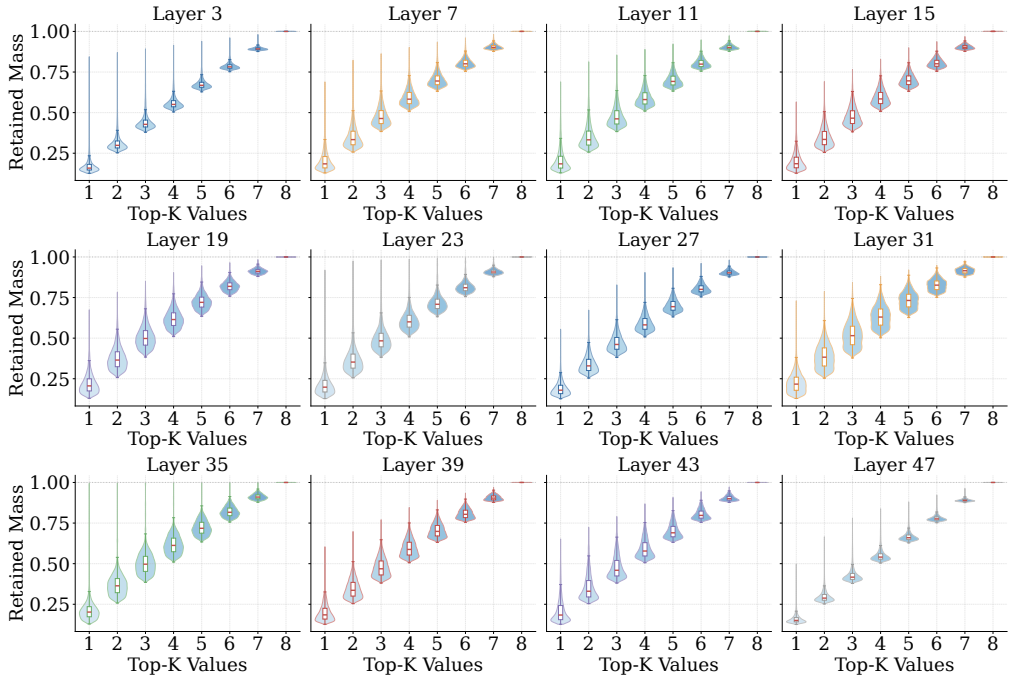

Figure 6: Distribution of retained routing mass across visual tokens across layers of Qwen3-VL-30B-A3B-Instruct. Retained routing mass is defined in Eq. equation 6.

Fig. 6 extends the token-level analysis in Fig. 1 *Right* to additional MoE layers spanning different depths of Qwen3-VL-30B-A3B-Instruct. Across these layers, visual tokens consistently exhibit a broad distribution of retained routing mass under the same Top-K value. In particular, some tokens concentrate most of their routing mass on only a small number of experts, whereas others require substantially more experts to retain a comparable fraction of routing mass.

The phenomenon is therefore not restricted to the two representative layers visualized in the main paper. Although the detailed distributions vary across layers, substantial within-layer token-level heterogeneity persists from earlier to later MoE layers. These results further motivate allocating each layer's expert budget adaptively among visual tokens instead of enforcing the same routing capacity for every token.

C Baseline Implementation Details

Ascend and Descend. Ascend and Descend use the same total visual-token routing budget as M^{3}MoE. Given a target budget, we first construct a smooth layer-wise Top-K distribution over $\{0, \ldots, K_{\max}\}$ such that the sum of Top-K values exactly equals the target budget. The layer counts for different Top-K values are kept approximately balanced, and neighboring capacity levels differ by at most one. Ascend places these Top-K values in non-decreasing order with layer depth, while Descend reverses the ordering. Therefore, both baselines have identical overall capacity distributions and differ only in whether larger routing capacities are assigned to shallower or deeper layers.

Alloc-MoE. Alloc-MoE was originally proposed for text-only MoE LLMs and adaptively allocates expert-activation capacity across both layers and tokens. To evaluate its transferability to multimodal MoE inference, we preserve its original allocation mechanisms while restricting capacity reduction to visual tokens. We use GQA as the calibration dataset for sensitivity profiling and follow the original allocation-isolated profiling strategy. At the token level, we follow the original Alloc-MoE allocation scheme and set $K_{\text{base}} = 1$ for the Average-K settings of $\bar{K} = 2$ and $\bar{K} = 1$, guaranteeing one expert activation for each visual token before distributing the remaining layer-wise budget according to routing scores. Under the extremely sparse setting of $\bar{K} = 0.5$, where such a minimum activation constraint would exceed the target budget, we instead set $K_{\text{base}} = 0$ to make sub-unit average routing capacity feasible. For fair comparison, the total visual-token expert budget is matched to the corresponding Average-K setting of M^{3}MoE, while the routing configuration of text tokens remains unchanged.

FastMMoE. We adopt the expert-activation reduction component of FastMMoE while disabling its routing-aware token pruning. Visual tokens before the reduction boundary l_v retain the default routing capacity $K_{\max}$, whereas subsequent layers use fewer routed experts selected by the largest routing probabilities; text tokens always retain the default routing capacity, and shared experts, when present, remain fully activated. To ensure exact budget matching, especially under the extremely sparse Average-$K = 0.5$ setting, we use a boundary-based allocation in which shallow layers keep $K_{\max}$, sufficiently deep layers use $K = 0$, and the entire residual budget is assigned to the boundary layer. This preserves FastMMoE's layer-wise reduction principle while matching the prescribed visual-token expert budget exactly.

MoDES. We adapt MoDES to reduce expert activations only for visual tokens, retaining its original scoring mechanism that combines global layer-wise importance with token-level expert importance. Using GQA for offline profiling, we collect the resulting visual-token expert scores and select a skipping threshold for each target Average-K to match the corresponding M^{3}MoE budget as closely as possible on the calibration set. During inference, the threshold remains fixed, and experts selected by the default router are skipped when their combined scores fall below it. Text-token routing remains unchanged. Although input-dependent skipping may cause the realized inference budget to deviate from the calibration target, the threshold is calibrated to keep the budget as close as possible to that of the corresponding M^{3}MoE setting.

Table 7: Performance on Qwen3-VL across different Average-K settings.

Method	TextVQA	ChartQA	MMStar	MMBench	MME	AI2D	DocVQA	OCRBench	Avg.(%) ↑
K=8 (Default)	83.57	84.84	60.02	86.94	2499	86.14	92.34	83.20	100.00
$\bar{K} = 2$ *(Skip 75% Visual Expert Activations)*									
Uniform	81.55	82.24	58.13	85.57	2419	84.33	91.21	76.60	96.92
Ascend	79.52	73.32	53.84	82.56	2130	81.61	86.86	75.60	91.39
Descend	80.44	77.12	58.03	85.57	2432	84.75	91.47	78.60	96.44
FastMMoE	78.70	71.00	55.54	84.79	2284	83.06	88.13	75.40	92.73
Alloc-MoE	81.76	80.36	57.36	85.57	2376	84.26	90.35	79.60	96.60
MoDES	81.25	80.20	58.40	85.74	2428	85.36	91.40	79.40	97.29
M^{3}MoE	**82.24**	**83.92**	**58.96**	**85.83**	**2463**	**85.52**	**92.32**	**82.20**	**98.86** (+1.57)
$\bar{K} = 1$ *(Skip 87.5% Visual Expert Activations)*									
Uniform	80.79	71.72	57.84	84.57	2269	83.68	90.04	75.60	93.90
Ascend	76.81	66.80	54.36	81.96	2135	80.89	83.27	72.10	88.96
Descend	77.77	69.64	54.86	83.42	2253	82.21	84.86	74.80	91.24
FastMMoE	75.98	66.92	54.09	82.99	2172	81.19	83.88	72.10	89.25
Alloc-MoE	80.89	77.44	55.98	85.22	2313	83.35	90.27	78.10	95.04
MoDES	80.68	78.48	57.35	84.51	2358	84.10	88.72	77.10	95.32
M^{3}MoE	**81.61**	**81.44**	**58.63**	**85.91**	**2421**	**85.04**	**92.64**	**79.70**	**97.73** (+2.41)
$\bar{K} = 0.5$ *(Skip 93.75% Visual Expert Activations)*									
Ascend	74.99	64.84	54.16	80.30	2115	80.93	80.89	70.90	87.52
Descend	75.98	67.20	53.97	82.73	2166	81.22	84.25	72.90	89.38
FastMMoE	74.99	65.65	53.61	82.39	2142	81.38	82.24	71.60	88.31
Alloc-MoE	79.24	73.60	55.65	84.28	2264	83.16	88.91	76.10	93.26
MoDES	77.16	71.68	56.38	83.51	2199	83.06	86.75	73.80	91.73
M^{3}MoE	**80.79**	**77.56**	**57.94**	**85.40**	**2399**	**84.78**	**91.07**	**78.40**	**96.27** (+3.01)

D Additional Results on Qwen3-VL

To further evaluate the generalizability of M^{3}MoE across different MoE-based MLLM architectures, we conduct the same comparison on Qwen3-VL. Tab. 7 reports the complete results under the same Average-K settings and routing-budget protocol used in the main experiments.

Consistent with the observations on Kimi-VL, M^{3}MoE achieves the highest average normalized performance across all evaluated capacity budgets. Specifically, it outperforms the strongest competing baseline by 1.57%, 2.41%, and 3.01% at $\bar{K} = 2$, 1, and 0.5, respectively. Even under the most aggressive routing capacity reduction at $\bar{K} = 0.5$, M^{3}MoE retains 96.27% of the default model performance.

The performance margin also increases as the routing budget becomes more constrained, exhibiting a trend consistent with that observed on Kimi-VL. Moreover, M^{3}MoE maintains its advantage over heuristic allocation strategies, Alloc-MoE, FastMMoE, and MoDES across the evaluated settings. These results provide additional evidence that the effectiveness of sensitivity-guided layer allocation and routing-weight-based token-level redistribution is not specific to a particular model architecture, but generalizes across different MoE-based MLLMs.

E Efficient Prefill Implementation

We apply the following system-level optimizations to translate the reduced MoE computation into actual prefill speedups. **(i) Sparse expert execution.** The layer-wise budget and the token–expert selection are materialized as routing masks, so experts that receive no tokens are never executed and their outputs are directly written as zeros, with no dispatch or kernel launch. **(ii) Lightweight routing.** A customized gate performs top-K selection only over the retained experts and writes the sparse weights in a single pass, avoiding the full score materialization and scatter of the default router, while remaining numerically equivalent to it. **(iii) Fused expert computation.** All activated experts of a layer are computed by two Triton grouped GEMM kernels (gate/up projection with SiLU

and down projection) plus one deterministic token-reduction kernel; the expert-to-tile metadata is built entirely on the GPU with no host synchronization.

F Detailed Results of Visual and Text Routing Reduction

To further analyze the sensitivity of different token types to routing capacity reduction, we report the complete per-benchmark results on Kimi-VL. For each reduced token type, its routing capacity is changed from the default $K = 6$ to $K = 1$, while the other token type retains the default routing capacity. This experiment is intended to compare the sensitivity of visual and text routing under the same per-token Top-K reduction, rather than to match the total number of expert activations across configurations.

Table 8: Per-benchmark results of routing-capacity reduction on different token types in Kimi-VL. The default routing capacity is $K = 6$. In "Visual only" and "Text only", only the corresponding token type is reduced to Top-1, while the other retains the default capacity. "Visual + Text" reduces both token types to Top-1. Bold denotes the best result among three settings.

Setting	TextVQA	ChartQA	MMStar	MMBench	MME	AI2D	DocVQA	OCRBench	Avg.(%)
Default ($K = 6$)	88.70	89.44	49.55	83.33	2201	83.61	97.78	83.60	100.00
Visual only ($K_v = 1$)	83.16	84.68	**47.12**	**81.69**	**2101**	**81.90**	94.17	74.50	**95.05**
Text only ($K_t = 1$)	**86.35**	**84.88**	46.26	60.40	1770	70.79	**96.60**	**83.10**	90.17
Visual + Text ($K_v = K_t = 1$)	83.14	81.20	45.36	59.19	1698	68.65	92.24	79.20	86.93

Reducing visual-token routing preserves substantially more overall performance than reducing text-token routing. In particular, the gap is large on MMBench, MME, and AI2D, while visual-only reduction remains close to the default model on most benchmarks. These results support our design choice of preserving the original routing capacity for text tokens and concentrating routing-capacity reduction on visual tokens.

G Dynamic-Programming Allocation

Since LSP provides the complete sensitivity profile $S_{l,k}$ for every MoE layer and candidate routing capacity, a natural alternative to SLA is to directly optimize the discrete layer-wise capacities using these profiles. We therefore construct a dynamic-programming (DP) allocator that searches for the capacity assignment with the minimum accumulated profiled sensitivity under the same global routing budget. This experiment examines whether directly optimizing the fine-grained LSP measurements yields a better layer-wise allocation than the aggregated sensitivity signal used by SLA.

Implementation Details. Given the LSP sensitivity profiles, the DP allocator solves

$$\begin{aligned} \min_{\{k_l\}_{l=1}^{L}} \quad & \sum_{l=1}^{L} S_{l,k_l} \\ \text{s.t.} \quad & \sum_{l=1}^{L} k_l = \mathcal{B}, \qquad 0 \le k_l \le K_{\max}, \end{aligned} \tag{8}$$

where $S_{l,K_{\max}} = 0$ and $\mathcal{B}$ denotes the target global routing-capacity budget. The DP state records the minimum accumulated sensitivity after assigning a given amount of routing capacity to the first several MoE layers. At each layer, all candidate capacities $k_l \in \{0, \dots, K_{\max}\}$ are considered, and the final layer-wise assignment is recovered by backtracking from the state corresponding to $\mathcal{B}$. Thus, DP obtains the exact optimum of the additive objective in Eq. equation 8. For a controlled comparison, DP and SLA use exactly the same LSP sensitivity profiles and the same global visual-token routing budgets. TAR is disabled for both methods so that the experiment isolates only the effect of layer-wise capacity allocation.

Table 9: Per-benchmark comparison between dynamic-programming allocation and SLA on Qwen3-VL. Both methods use the same LSP sensitivity profiles and visual-token routing budgets, with TAR disabled. The better result between the two allocation strategies is highlighted in bold for each budget.

Method	TextVQA	ChartQA	MMStar	MMBench	MME	AI2D	DocVQA	OCRBench	Avg.(%) ↑
$K = 8$ *(Default)*	83.57	84.84	60.02	86.94	2499	86.14	92.34	83.20	100.00
$\bar{K} = 2$ *(Skip 75% Visual Expert Activations)*									
DP	**82.41**	**82.84**	**59.29**	85.65	2407	85.14	92.17	81.70	98.34
SLA	81.92	82.72	58.88	**86.43**	**2468**	**85.72**	**92.30**	**82.10**	**98.74** (+0.40)
$\bar{K} = 1$ *(Skip 87.5% Visual Expert Activations)*									
DP	**81.46**	**80.56**	57.58	84.36	2308	84.16	91.24	79.20	96.18
SLA	81.16	80.40	**58.57**	**85.40**	**2411**	**84.88**	**91.84**	**80.00**	**97.29** (+1.11)
$\bar{K} = 0.5$ *(Skip 93.75% Visual Expert Activations)*									
DP	79.72	76.12	55.85	83.08	2151	82.25	89.49	76.30	92.99
SLA	**80.31**	**76.76**	**56.87**	**85.48**	**2323**	**84.16**	**90.85**	**77.50**	**95.23** (+2.24)

Results and Discussion. As shown in Tab. 9, SLA consistently achieves higher normalized average performance than the DP allocator across all three routing budgets. The margin increases as the routing capacity becomes more constrained, from +0.40% at $\bar{K} = 2$ to +1.11% at $\bar{K} = 1$ and +2.24% at $\bar{K} = 0.5$. The per-benchmark results reveal the same trend. SLA outperforms DP on five of the eight benchmarks at $\bar{K} = 2$ and six at $\bar{K} = 1$. Under the most aggressive setting, $\bar{K} = 0.5$, SLA achieves higher performance on all eight benchmarks. Therefore, the growing advantage under aggressive sparsity is broadly distributed across tasks rather than being dominated by a single benchmark.

Interestingly, DP is the exact optimizer of the additive objective defined by the independently profiled sensitivities, yet its resulting allocation is consistently inferior to SLA in end-to-end evaluation. This discrepancy suggests that the isolated sensitivity measurements $S_{l,k}$ are not strictly additive when routing capacities are reduced simultaneously across multiple layers. In particular, interactions among layer-wise perturbations can cause an allocation that is optimal under the summed profiling objective to become suboptimal after all reductions are applied jointly.

SLA instead compresses each layer's sensitivity profile into a layer-level importance signal and performs proportional budget allocation. Although this does not explicitly optimize the discrete additive objective, the aggregated signal appears less sensitive to errors introduced by the independence assumption in layer-wise profiling. The increasingly large gap under smaller routing budgets further suggests that such cross-layer interactions become more consequential as sparsity becomes more aggressive.