Alloc-MoE: Budget-Aware Expert Activation Allocation for Efficient Mixture-of-Experts Inference

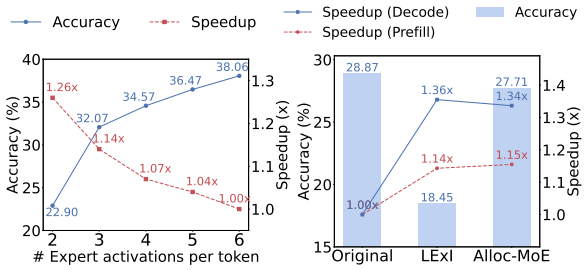
Baihui Liu, Kaiyuan Tian, Wei Wang, Zhaoning Zhang*, Linbo Qiao, Dongsheng Li
National Key Laboratory of Parallel and Distributed Computing,
College of Computer Science and Technology,
National University of Defense Technology
{lbh,kyt,wwking,zhangzhaoning,qiao.linbo,dsli}@nudt.edu.cn

Abstract

Mixture-of-Experts (MoE) has become a dominant architecture for scaling large language models due to their sparse activation mechanism. However, the substantial number of expert activations creates a critical latency bottleneck during inference, especially in resource-constrained deployment scenarios. Existing approaches that reduce expert activations potentially lead to severe model performance degradation. In this work, we introduce the concept of *activation budget* as a constraint on the number of expert activations and propose Alloc-MoE, a unified framework that optimizes budget allocation coordinately at both the layer and token levels to minimize performance degradation. At the layer level, we introduce Alloc-L, which leverages sensitivity profiling and dynamic programming to determine the optimal allocation of expert activations across layers. At the token level, we propose Alloc-T, which dynamically redistributes activations based on routing scores, optimizing budget allocation without increasing latency. Extensive experiments across multiple MoE models demonstrate that Alloc-MoE maintains model performance under a constrained activation budget. Especially, Alloc-MoE achieves $1.15\times$ prefill and $1.34\times$ decode speedups on DeepSeek-V2-Lite at half of the original budget.

1 Introduction

Mixture-of-Experts (MoE) has emerged as an important approach for sparsifying mainstream Transformer-based models (Shazeer et al., 2017; Lepikhin et al., 2021; Fedus et al., 2022). It replaces the feed-forward network layers with sparse MoE layers, which consist of a gating network and a set of small networks named *experts*. The gating network computes routing scores over experts per token and activates only the Top-K experts for each token. This sparse expert activation mechanism facilitates widespread deployment of MoE models in real-world systems (DeepSeek-AI, 2024b; Yang et al., 2025a; Team et al., 2025).



(a) (b)

Figure 1: (a) Reducing the number of expert activations per token increases speedup but decreases accuracy. (b) Alloc-MoE achieves inference latency comparable to the mainstream expert activation reduction method while retaining performance close to that of the original full-activation model when halving expert activation per token. All results are measured on DeepSeek-V2-Lite. Speedup is computed as the inference latency of the original model (with 6 expert activations per token) divided by the latency of the evaluated method.

However, as the number of input tokens increases, the large amount of expert activations becomes a critical bottleneck for efficient MoE inference, which is further severe in resource-constrained deployment scenarios. Existing works have explored reducing expert activations either from token-level (Yang et al., 2024; Huang et al., 2024; Guo et al., 2025; Aghdam et al., 2024; Lu et al., 2024; Muzio et al., 2024; Zhong et al., 2024; Huang et al., 2025) or layer-level (Yang et al., 2025b; Chitty-Venkata et al., 2025) perspective to decrease inference latency. However, these approaches neglect their impact on model performance, potentially leading to significant degradation. Figure 1(a) shows that on DeepSeek-V2-Lite, reducing the expert activations per token from 6 to 3 leads to a 17% performance degradation, which exacerbates to nearly 40% when further reduced to

*Corresponding Author.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 9653–9667
July 2-7, 2026

two activated experts.

To address this problem, we formalize the number of expert activations as an *activation budget*, which is closely correlated with inference latency, and propose *Alloc-MoE*, a unified framework that optimizes budget allocation to minimize performance degradation under a fixed expert activation budget. Figure 1(b) demonstrates that Alloc-MoE achieves a comparable speedup to the mainstream method and maintains performance close to the original model. Alloc-MoE coordinately allocates the budget at the layer and token levels utilizing *Alloc-L* and *Alloc-T*, respectively. Alloc-L adopts an end-to-end performance metric to profile layer sensitivity and formulates layer-level expert activation allocation as a sensitivity-guided optimization problem, which is solved exactly and efficiently via dynamic programming, yielding the optimal allocation of expert activations across layers. Building upon this, Alloc-T dynamically redistributes expert activations across tokens within each layer according to token-level routing scores. By prioritizing tokens with less concentrated routing distributions, Alloc-T better allocates limited expert activations without introducing extra inference latency.

Our contributions are summarized as follows:

- We introduce the expert activation budget and propose Alloc-MoE, a unified framework that optimizes the allocation of budgets coordinately at the layer and token levels.
- We present Alloc-L, a layer-level expert activation allocation method that optimizes layer-level expert activation allocation under a fixed budget by leveraging global sensitivity profiling and exact dynamic programming.
- We introduce Alloc-T, a token-level redistribution strategy that reallocates expert activations according to routing scores, improving model performance under a fixed activation budget without additional inference latency.
- Extensive experiments demonstrate that Alloc-MoE sustains performance under a restricted activation budget across multiple MoE models. Notably, on DeepSeek-V2-Lite, it attains $1.15\times$ speedup in prefill and $1.34\times$ in decode when using only half of the original budget.

2 Related works

Token-level Expert Activation Reduction. These methods aim to adaptively reduce the number of activated experts per token based on token-level routing scores. XMoE (Yang et al., 2024) introduces Top-P routing, where each token activates a variable number of experts whose cumulative routing scores exceed a predefined threshold P. However, it requires training-time calibration and can lead to over-activation. Dynamic-MoE (Huang et al., 2024) further regularizes the entropy of routing score distributions during training to mitigate the over-activation behavior of Top-P routing. NAEE (Lu et al., 2024) conditionally skips the secondary expert in Top-2 routing for Mixtral (Jiang et al., 2024) when its gating weight falls below a relative threshold, but this approach is limited to models with Top-2 routing and does not generalize to larger expert sets. AdapMoE (Zhong et al., 2024) employs a sensitivity-aware gating mechanism with an offline-calibrated threshold to adaptively reduce expert activations while preserving model performance, yet it is similarly constrained to specific pretrained models and requires offline calibration.

Layer-level Expert Activation Allocation. These approaches typically reduce the number of per-layer expert activations and demonstrate the efficiency improvement at the cost of performance degradation. Yang et al. (2025b) investigate several heuristic expert reduction strategies, showing significant throughput improvements across both low- and high-concurrency settings, and revealing that their efficiency–accuracy trade-offs differ markedly across MoE models. LExI (Chitty-Venkata et al., 2025) proposes a data-free, post-training layer-adaptive reduction method and demonstrates improved inference efficiency compared to pruning-based approaches.

3 Methodology

In this section, we first review the Top-K routing mechanism and formalize the notion of expert activation budget. Then we present the two key components of Alloc-MoE: Alloc-L for layer-level activation allocation and Alloc-T for token-level activation redistribution.

3.1 Preliminary

Top-K Routing of Mixture-of-Experts. A standard MoE layer consists of N expert networks and a gating network that enables conditional computation by selectively activating a subset of experts

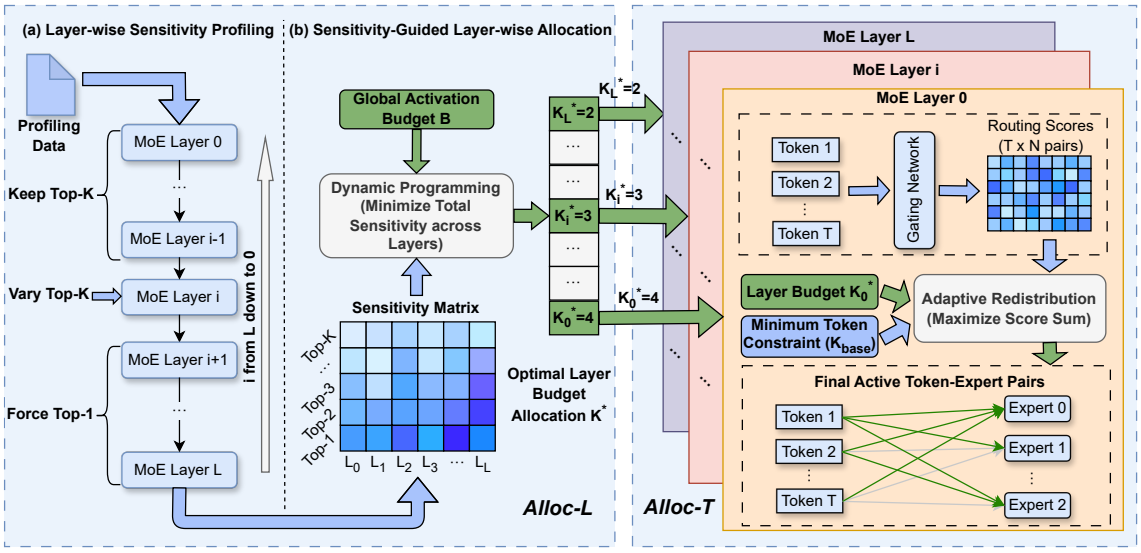


Figure 2: Overview of the Alloc-MoE framework, consisting of two components: **Alloc-L** (left) profiles layer-wise sensitivity to construct a sensitivity matrix and applies dynamic programming to determine the optimal layer-level activation budget $\mathbf{K}^*$ under a global expert activation budget B; **Alloc-T** (right) adaptively redistributes expert activations across tokens based on routing scores, maximizing the total routing weight while respecting the layer-wise budgets.

for each token. Given an input hidden representation $X \in \mathbb{R}^{T\times H}$, the gating network independently routes each token x_t by projecting it with $W_{\text{gate}} \in \mathbb{R}^{H\times N}$ and computing routing scores over experts:

$$w_t = \text{Top-K}(\text{Softmax}(x_t \cdot W_{\text{gate}})), \qquad (1)$$

where only the K largest scores are retained and the rest are masked to zero. The MoE output is computed as a weighted sum over the activated experts,

$$y_t = \sum_{i=1}^{N} w_{t,i} \cdot E_i(x_t), \qquad (2)$$

with $w_{t,i} = 0$ for non-activated experts.

Expert Activation Budget. In an MoE layer, each activated expert corresponds to one feed-forward execution, typically implemented as a GEMM operation. As a result, the inference latency scales approximately linearly with the total number of expert activations. We formalize expert activations using two closely related budget notions: a *global activation budget* and a *layer-level activation budget*. Specifically, we define the *global activation budget* as the total number of expert activations incurred by a single token across all MoE layers during inference. For a model with L MoE layers, if a token activates up to Top-K experts in each layer, its global activation budget is bounded by $L \times K$. We define the *layer activation budget* as the average number of expert activations allocated per token within a given MoE layer. For a layer processing T tokens with a Top-K routing, the total activation budget is at most $T \times K$, which can be flexibly redistributed across tokens as long as the average per-token budget is preserved. The *global activation budget* equals the sum of the *layer activation budgets* across all layers. This formalization enables principled allocation at both the layer and token levels, directly motivating the design of Alloc-L and Alloc-T.

3.2 Alloc-L

Alloc-L optimizes the allocation of expert activations across layers under a fixed global activation budget. It profiles layer-wise sensitivity using an end-to-end performance metric and leverages this information to perform a sensitivity-aware allocation, which can be solved efficiently via dynamic programming.

Layer-wise Sensitivity Profiling. The sensitivity of expert allocations in shallow layers may be masked by compensatory allocations in deeper layers, obscuring the attribution of observed performance changes to a specific layer. To better characterize the layer-wise sensitivity, we adopt an allocation-isolating profiling strategy that mitigates interference from subsequent layers.

Specifically, we utilize the perplexity metric. For a model with L MoE layers indexed $\{0, 1, \ldots, L-1\}$, when profiling target layer i, we gradually reduce its allocated Top-K value from the original K_{orig} down to 1, while temporarily constraining all deeper layers to the minimal activation setting (Top-$K = 1$) and keeping all preceding layers at the original K_{orig}. This preserves routing patterns in preceding layers and mitigates compensatory effects from deeper layers, enabling accurate attribution of observed performance changes to the target layer. The perplexity measured under different Top-K settings is then used to characterize the relative sensitivity of target layer i to changes in expert activation.

After applying the profiling process in Algorithm 1, we obtain a global sensitivity matrix $S \in \mathbb{R}^{L \times K_{\text{orig}}}$, where each normalized row $S[i, 1..K_{\text{orig}}]$ captures the relative loss of layer i under varying expert activations and provides a globally comparable layer-wise sensitivity measure across layers.

Sensitivity-Guided Layer-wise Expert Allocation. Consider a model with L MoE layers, let B denote the maximum global budget, K_{orig} the original Top-K of the model, and define $\mathbf{K} = [K_0, K_1, \ldots, K_{L-1}]$ as a layer-level activation budget allocation, where $1 \leq K_i \leq K_{\text{orig}}$ represents the budget of expert activations per token allocated to layer i. Our goal is to identify the optimal layer-level allocation $\mathbf{K}^*$ that minimizes the aggregated loss under the global activation budget:

$$\underset{\mathbf{K}=[K_i]_{i=0}^{L-1}}{\operatorname{argmin}} \quad \sum_{i=0}^{L-1} \mathbf{S}[i, K_i] \tag{3}$$

$$\text{s.t.} \quad \sum_{i=0}^{L-1} K_i \leq B, \tag{4}$$

$$1 \leq K_i \leq K_{\text{orig}}, \quad \forall i. \tag{5}$$

We solve this problem by casting it as a budget-constrained allocation task analogous to a grouped knapsack problem. Each MoE layer constitutes a group of allocation choices, where activating k experts per token for layer i consumes a budget of k and incurs a sensitivity cost of $\mathbf{S}[i, k]$. The objective is to minimize the total sensitivity across all layers under the global budget constraints. We define $\mathrm{DP}[i, b]$ as the minimum cumulative sensitivity achievable by allocating experts to layer $0, 1, \cdots, i$ under a total budget b. The recurrence relation is:

$$\mathrm{DP}[i, b] = \min_{k \leq b} \left(\mathrm{DP}[i-1, b-k] + \mathbf{S}[i, k]\right) \tag{6}$$

with base conditions $\mathrm{DP}[-1, 0] = 0$ and $\mathrm{DP}[-1, b > 0] = +\infty$.

After processing all L layers, the optimal allocation is obtained as $\mathbf{K}^* = \underset{b \leq B}{\operatorname{argmin}}\, \mathrm{DP}[L-1, b]$, with the corresponding layer-wise expert allocation recovered via backtracking. This dynamic programming formulation yields an exact solution with complexity $O(L \cdot B \cdot K_{\text{orig}})$, which is efficient in practice due to the limited allocation range per layer, as the total budget B is upper-bounded by $L \times K_{\text{orig}}$.

Algorithm 1: Layer-wise Allocation-Isolated Sensitivity Profiling

Input: Model M, number of layers L, calibration dataset D_{calib}, original Top-K K_{orig}
Output: Global sensitivity matrix $S \in \mathbb{R}^{L \times K_{\text{orig}}}$

```
Initialize S ← 0, C ← [K_orig]_{i=0}^{L-1}
ApplyConfiguration(M, C)
PPL ← GetPerplexity(M, D_calib)
for i = L − 1 down to 0 do
    S[i, K_orig] ← PPL
    for k = K_orig − 1 down to 1 do
        C[i] ← k
        ApplyConfiguration(M, C)
        PPL ← GetPerplexity(M, D_calib)
        S[i, k] ← PPL
    end for
end for
return S
```

3.3 Alloc-T

Alloc-T optimizes the allocation of expert activations across tokens within each layer under a fixed layer-level activation budget. It leverages token-level routing score distributions to adaptively redistribute activations, improving allocation without increasing the overall inference cost.

Token-level Adaptive Expert Activation Redistribution. Alloc-T treats expert activation allocation within a layer as a collective allocation problem across tokens under a fixed layer-level activation budget, enabling expert activation to be adaptively allocated to tokens according to their routing score distributions.

Specifically, let K_l denote the average activation budget per token for layer l as determined by Alloc-L. We define the candidate set of token–expert pairs as

$$\mathcal{C}_l = \{(t, e) | e \in \text{Top-K}(w_t), t = 1, \ldots, T\}, \quad (7)$$

where T is the number of tokens and $|\mathcal{C}_l| = T \times K_l$. Alloc-T then selects activations from $\mathcal{C}_l$ while respecting the average activation budget K_l.

We introduce binary variables $z_{t,e} \in \{0, 1\}$ to indicate whether expert e is activated by token t, and formulate token-level expert activation allocation as:

$$\max_{z_{t,e}} \quad \sum_{(t,e)\in\mathcal{C}_l} z_{t,e} \cdot w_{t,e} \quad (8)$$

$$\text{s.t.} \quad \sum_{(t,e)\in\mathcal{C}_l} z_{t,e} \leq T \times K_l, \quad (9)$$

$$\sum_{e} z_{t,e} \geq K_{\text{base}}, \quad \forall t. \quad (10)$$

Constraint 9 enforces the layer-level activation budget at layer l, while Constraint 10 ensures a minimum base expert activation allocation K_{base} per token to prevent token dropping. In practice, this constrained optimization is efficiently solved by using simple masking and global top-selection operations on the routing scores, incurring negligible inference overhead even at large T.

Specifically, given the routing score *scores* of shape $[T, K_{\text{orig}}]$, each row $scores[i, :]$ contains the routing scores for token i, sorted in descending order. We first preserve the Top-K_{base} experts for each token to ensure routing stability and maintain a minimum allocation. Then, instead of performing independent per-token selection, we collect the remaining $K_{\text{orig}} - K_{\text{base}}$ candidate expert scores across all tokens and globally select the top $(K_l - K_{\text{base}}) \cdot T$ entries under the overall activation budget constraint. This global selection enables computation to be dynamically shifted toward tokens with less concentrated routing distributions.

Notably, standard Top-K routing is a special case of this formulation when $K_{\text{base}} = K_l$ and no additional budget is available for redistribution. Alloc-T therefore generalizes conventional routing by relaxing fixed expert activation allocation per token and enabling adaptive token-level expert activation allocation under a fixed layer-level activation budget.

Model	# MoE Layers	# Act. / Tot. Experts	# Act. / Tot. Params.
DeepSeek	26	6 / 64	2.4B / 15.7B
Qwen	24	4 / 60	2.7B / 14.3B
OLMoE	16	8 / 64	1.0B / 7.0B

Table 1: Architectural details of the evaluated MoE models.

4 Experiments

4.1 Setup

Models and Budgets. We evaluate Alloc-MoE on three representative MoE models: DeepSeek-V2-Lite (DeepSeek-AI, 2024a), Qwen1.5-MoE-A2.7B (Team, 2024), and OLMoE-1B-7B-0924 (Muennighoff et al., 2025). These models differ in scale, number of MoE layers and Top-K configurations, providing a diverse evaluation benchmark for studying expert activation allocation strategies. For brevity, we refer to them as DeepSeek, Qwen, and OLMoE respectively in the following experiments. Table 1 summarizes their architectures.

For each model, we impose a strict global activation budget B. Specifically, we evaluate DeepSeek with $B \in \{130, 104, 78, 52\}$, corresponding to average per-layer Top-K allocations of $\{5, 4, 3, 2\}$. For Qwen, we consider $B \in \{84, 72, 60, 48\}$, corresponding to $\{3.5, 3, 2.5, 2\}$. For OLMoE, we adopt $B \in \{112, 96, 80, 64\}$, corresponding to $\{7, 6, 5, 4\}$. These budgets span mild to aggressive sparsification regimes, enabling systematic evaluation of performance–efficiency trade-offs under constrained expert activation allocation.

Baselines. We compare Alloc-MoE against representative MoE inference baselines that differ in how a fixed global activation budget is allocated across layers and tokens. Specifically, we consider: *Uniform*, which allocates identical expert activation to all MoE layers; *LExI*, which redistributes the budget across layers based on intra-layer sensitivity profiling; and two token-level baselines, *Dynamic-MoE* and *NAEE*, which adapt expert activation across tokens under the *Uniform* layer-wise allocations. Implementation details of layer- and token-level baselines are provided in Appendix A.1.

Datasets and Benchmarks. We use Wikitext2 (Merity et al., 2017) for calibration and con-

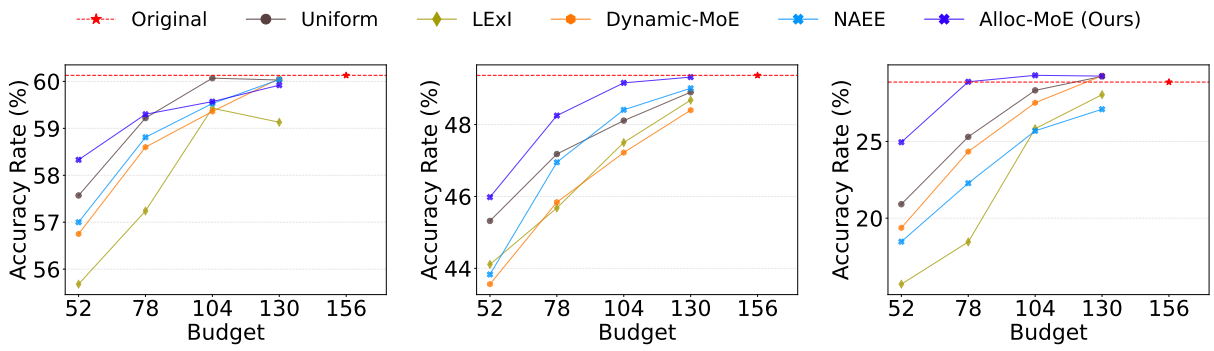


Figure 3: Alloc-MoE Results on DeepSeek-V2-Lite under varying global activation budgets. (a) NLU task, (b) Reasoning task and (c) Math task.

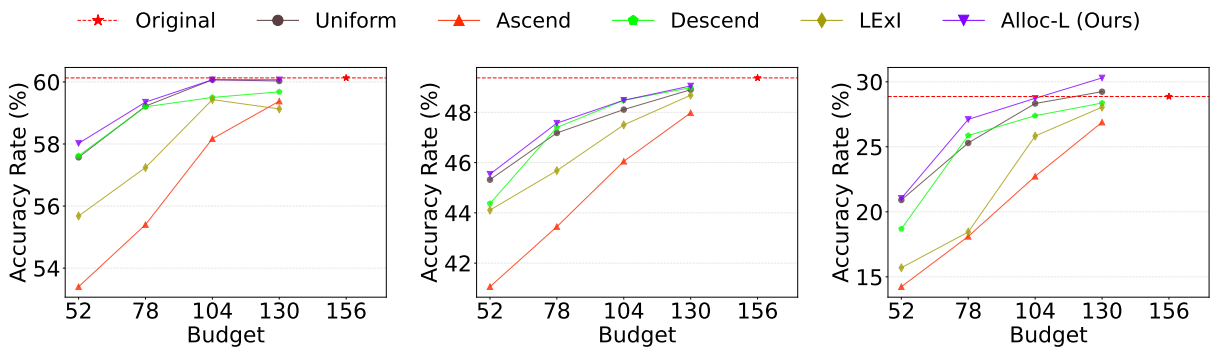
duct extensive evaluations on 20 datasets covering three task groups: Natural Language Understanding (NLU), Reasoning and Math. The NLU benchmarks comprise BoolQ (Clark et al., 2019), LAMBADA (Paperno et al., 2016), RACE (Lai et al., 2017), SciQ (Welbl et al., 2017), MNLI, QNLI and RTE (Wang et al., 2018). The Reasoning tasks include ARC (ARC-E and ARC-C) (Clark et al., 2018), HellaSwag (Zellers et al., 2019), LogiQA (Liu et al., 2021), MMLU (Hendrycks et al., 2021), PIQA (Bisk et al., 2020), TruthfulQA (Lin et al., 2022), ACP (Kokel et al., 2025), BBH (Suzgun et al., 2023), GroundedCocoa (Kohli et al., 2025), and SWAG (Zellers et al., 2018). The Math benchmarks include GSM8K (Cobbe et al., 2021), ASDiv (Miao et al., 2020), and MathQA (Amini et al., 2019). For GSM8K, ACP, and BBH, we report exact match (EM) metric. Accuracy metric is used for all other benchmarks. For clarity, we report the average performance within each task group.

Evaluation details. For performance evaluation, we use the `lm-eval` (Gao et al., 2024) framework with vllm (Kwon et al., 2023) backend. For inference efficiency evaluation, we report prefill and decode speedups relative to the original expert activation allocation strategy, using the DeepSeek model as a representative benchmark. *LExI* allocation under the same global activation budgets is included as a reference. Implementation and measurement details are provided in Appendix A.2. All experiments are conducted on a single NVIDIA H100 80GB GPU.

4.2 Main Results

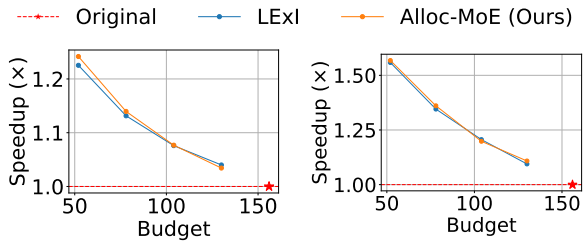
Performance. Figure 3 shows task-aggregated performance across varying budgets. From these results, several observations can be drawn: (1) Alloc-MoE outperforms baselines in 10 of 12 evaluated settings, demonstrating robust generalization across tasks and budgets. (2) As the budget becomes increasingly restrictive, Alloc-MoE exhibits minimal performance degradation relative to all baselines, underscoring its robustness to aggressive expert sparsification. (3) The performance advantage of Alloc-MoE grows with task complexity, with average improvements of 0.05% on NLU, 0.70% on Reasoning, and 2.15% on Math tasks. This indicates that Alloc-MoE is more beneficial for tasks with greater computational diversity, where adaptive activation allocation can better match the varying demands across tokens and layers, leading to more pronounced improvements compared to *Uniform* allocation. Similar trends are observed on Qwen and OLMoE in Appendix B (Figure 9), Alloc-MoE consistently maintains competitive performance under mild budget constraints and demonstrates increasingly clear advantages under aggressive sparsification, particularly on Reasoning and Math tasks. Overall, these results validate Alloc-MoE as a robust and general expert activation allocation framework for MoE inference under constrained budgets.

Inference Efficiency. As shown in Figure 5, Alloc-MoE achieves inference speedups comparable to *LExI* baseline across all global activation budgets in both prefill and decode stages, indicating that it introduces no additional runtime overhead. Moreover, inference latency decreases monotonically as the budget is reduced, suggesting that the observed speedups mainly stem from reduced expert activations. Under a representative setting where the budget is halved, Alloc-MoE achieves a $1.15\times$ speedup in prefill and a $1.34\times$ speedup in



(a) (b) (c)

Figure 4: Ablation results of Alloc-L on DeepSeek-V2-Lite under varying global activation budgets. (a) shows NLU task, (b) Reasoning task and (c) Math task.



(a) (b)

Figure 5: Speedup ratios for (a) prefill and (b) decode stages under varying global activation budgets.

K_{base}	Budget				Avg.
	52	78	104	130	
0	42.68	45.36	**45.99**	46.09	45.03
1	**42.83**	**45.42**	45.93	**46.30**	**45.12**
2	41.13	45.02	45.83	46.17	44.54
3	41.13	43.77	45.84	46.08	44.21
4	41.13	43.77	45.52	45.80	44.06
5	41.13	43.77	45.52	45.97	44.10

Table 2: Analysis of K_{base} in Alloc-T on DeepSeek-V2-Lite under *Uniform* allocation strategy. Results are averaged across three task groups, with the last column showing the average performance across all budgets.

decode relative to the original expert activation allocation strategy, demonstrating consistent inference acceleration under constrained budgets.

4.3 Analysis of the Base Expert Allocation K_{base} in Alloc-T

Based on *Uniform* allocation, we vary K_{base} from 0 to $K-1$ where K denotes the original Top-K of each model. Notably, $K_{base}=0$ corresponds to no base expert activation allocation, where all activated experts are selected solely through expert activation redistribution. To evaluate the overall impact of K_{base}, we report the average accuracy across three task groups.

Table 2 summarizes the results on DeepSeek. Across most budget settings, $K_{\text{base}}=1$ achieves the best or near-best performance and produces the highest average accuracy across all four budgets. In contrast, larger base allocations ($K_{base}\geq 2$) consistently degrade performance, especially under tighter budgets, suggesting that excessive base allocation over-constrains the allocation space and limits the effectiveness of adaptive expert allocation. While $K_{base}=0$ performs competitively, it remains slightly inferior to $K_{base}=1$, which indicates that $K_{\text{base}}=1$ consistently offers the most favorable trade-off between base allocation and flexible redistribution. Similar trends are observed on Qwen and OLMoE in Appendix B (Tables 5 and 6). Based on these observations, we set $K_{\text{base}}=1$ as the default configuration of Alloc-T.

4.4 Analysis of Expert Load Balance

Figure 7 illustrates the result of expert load distributions. Qualitatively, Alloc-MoE preserves the overall shape of the load distribution, introducing no noticeable distortion. Moreover, by reducing the per-expert load, it decreases the communication volume, which is expected to improve inference efficiency in distributed MoE deployment. We further conduct a quantitative analysis to characterize this behavior, as summarized in Figure 8. Specifically, we compute the *Spearman rank correlation* of expert loads across all layers between the two settings. The correlation remains consistently high (0.93–0.99), indicating that the relative ordering of *hot* and *cold* experts is largely preserved. This suggests that Alloc-MoE does not disrupt the inherent expert specialization. To further assess distributional shifts, we measure the difference in normal-

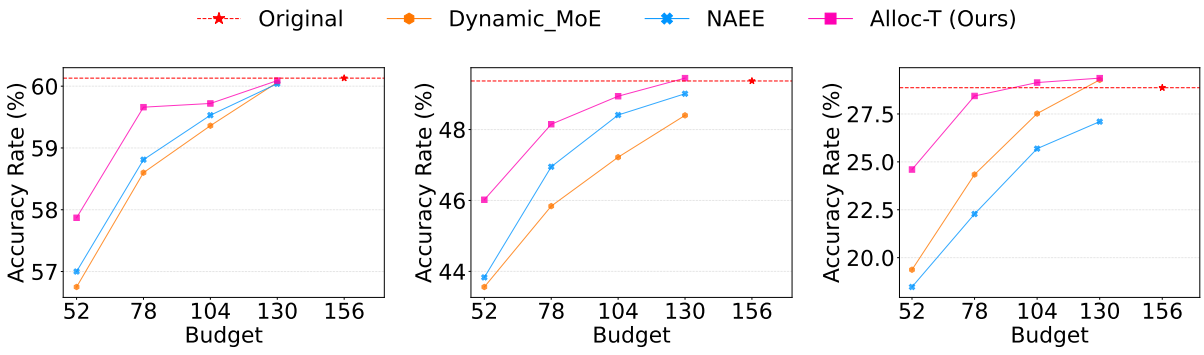


Figure 6: Ablation results of Alloc-T on DeepSeek-V2-Lite under varying global activation budgets. (a) shows NLU task, (b) Reasoning task and (c) Math task.

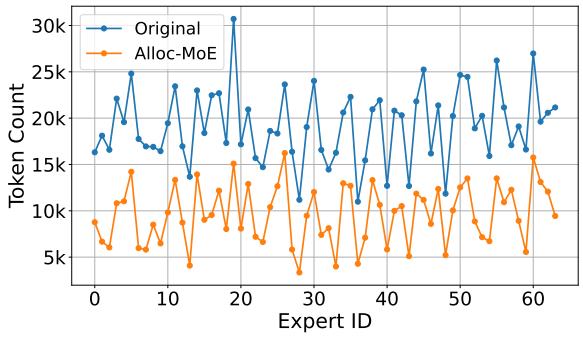


Figure 7: Expert load distributions in the 10-th MoE layer of DeepSeek-V2-Lite under the original configuration and Alloc-MoE with Budget = 78. Alloc-MoE preserves the load pattern while reducing per-expert load, potentially lowering inter-device communication overhead.

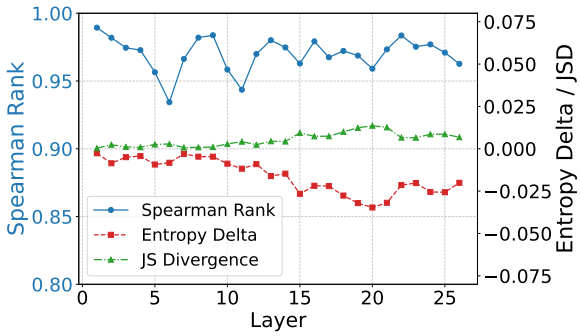


Figure 8: Analysis of load balance across layers. Including Spearman rank correlation (blue, left axis), Entropy delta (red, right axis), and JS divergence (green, right axis).

ized entropy between the two settings, as well as the Jensen–Shannon (JS) divergence between the corresponding weighted load distributions. Both metrics show minimal deviation: the entropy decreases slightly (0.003–0.035), while the JS divergence remains below 0.014 across all layers. These results demonstrate that Alloc-MoE maintains a stable expert utilization pattern while introducing negligible distributional shift.

4.5 Analysis of Budget Allocation in Alloc-L and Alloc-T

Under the half-budget setting, As shown in Appendix B (Figure 12), Alloc-L produces a clearly non-uniform layer-wise allocation, where earlier layers retain more experts while deeper layers operate with reduced budgets, reflecting heterogeneous routing sensitivity across layers. At the token level, Alloc-T shows strong correlation (> 0.7) with routing entropy and exhibits a consistent monotonic pattern, allocating fewer experts to low-entropy (high-confidence) tokens and more to high-entropy (ambiguous) ones.

Dataset	Budget				Avg.
	52	78	104	130	
WikiText2	41.53	44.61	45.76	46.47	44.59
C4	41.56	45.05	45.83	45.94	44.60
Pile	42.55	45.05	45.29	45.84	44.68

Table 3: Impact of different calibration datasets on DeepSeek-V2-Lite with Alloc-L under varying budgets.

4.6 Analysis of Calibration Dataset

We compare the impact of different calibration datasets, including WikiText2 (Merity et al., 2017), C4 (Raffel et al., 2020), and Pile (Gao et al., 2020). As shown in Appendix B (Table 3), the overall performance remains highly consistent across all three datasets under varying budget settings, indicating that Alloc-L is insensitive to the selection of calibration datasets.

Method	Budget				Avg.
	52	78	104	130	
Uniform	41.27	43.90	45.51	46.06	44.19
+L	41.53	44.61	45.76	**46.47**	44.59
+T	42.83	45.42	45.93	46.30	45.12
+L +T	**43.09**	**45.48**	**46.01**	46.17	**45.19**

Table 4: Ablation study of Alloc-L (**+L**) and Alloc-T (**+T**) on DeepSeek-V2-Lite under varying global activation budgets. Results are averaged across three task groups, with the last column showing the average performance across all budgets.

4.7 Ablation Studies

We first evaluate Alloc-L and Alloc-T independently against their respective level-specific baselines to demonstrate their individual effectiveness. We then analyze their joint behavior under the *Uniform* allocations to highlight their complementarity.

Effect of Alloc-L. We compare Alloc-L against four layer-wise allocation baselines under four budget settings. In addition to *LExI* and *Uniform*, we consider: *Ascending*, where the allocations increase with layer depth, and *Descending*, which applies the reverse schedule. Figure 4 presents the results on DeepSeek. Compared to these baselines, Alloc-L consistently achieves a superior performance–efficiency trade-off. Similar trends are observed on Qwen and OLMoE in Appendix B (Figure 10). Across both models, Alloc-L outperforms the baseline strategies in the majority of budget configurations, indicating that Alloc-L generalizes well across different MoE models.

Effect of Alloc-T. We compare Alloc-T against *NAEE* and *Dynamic-MoE* under the *Uniform* layer-wise allocations. Figure 6 presents the results on DeepSeek. Alloc-T consistently outperforms both baselines across all tasks and budgets. Notably, its advantage becomes increasingly pronounced under tighter budgets and on more challenging tasks, highlighting the benefit of token-wise redistribution under aggressive sparsification. Similar trends are observed on Qwen and OLMoE as shown in Appendix B (Figure 11), indicating that Alloc-T generalizes well across different MoE models.

Complementarity of Alloc-L and Alloc-T. Table 4 reports the ablation results on DeepSeek. The results show that: (1) Alloc-L consistently improves performance across all budget settings, achieving an average gain of 0.4%. (2) Alloc-T yields larger improvements than Alloc-L in most budget settings, particularly under aggressive sparsification, highlighting the growing importance of token-level redistribution as the budget becomes increasingly constrained. (3) Combining Alloc-L and Alloc-T consistently achieves the best or near-best performance across all budgets, with the highest average performance. Similar trends are observed on Qwen and OLMoE in Appendix B (Tables 7 and 8), indicating that the benefits of Alloc-L and Alloc-T generalize across different MoE models. Overall, these results suggest that Alloc-L and Alloc-T operate on orthogonal dimensions of expert activation allocation and can be jointly applied to improve the allocation of limited budgets.

5 Conclusion

In this work, we proposed Alloc-MoE, a unified framework that optimizes the allocation of limited expert activation in Mixture-of-Experts models to minimize performance degradation. By modeling expert activations as a global activation budget and allocating it in a coordinated manner across layers and tokens, Alloc-MoE effectively mitigates performance degradation caused by reduced expert activations. Extensive experiments across multiple MoE models and tasks demonstrate that Alloc-MoE consistently achieves a superior performance–efficiency trade-off, preserving accuracy even with substantially fewer activated experts.

Limitations

While Alloc-MoE demonstrates strong performance under constrained expert activation budgets, several limitations remain. First, while our approach focuses on allocating expert activations, it is fully orthogonal to other efficiency-oriented methods such as expert pruning or quantization, which could be combined with Alloc-MoE for additional speedups. Second, Alloc-MoE does not incorporate hardware-level factors such as expert placement or communication overhead. Integrating these considerations in distributed systems represents a natural direction for future enhancement. Finally, our framework targets pretrained models, and extending Alloc-MoE to incorporate activation-aware strategies during training to improve model robustness remains an open avenue for further research.

Acknowledgments

This work is sponsored in part by the National Natural Science Foundation of China (No. 62421002) and the Fundamental and Interdisciplinary Disciplines Breakthrough and Plan of the Ministry of Education of China (JYB2025XDXM202).

References

Maryam Akhavan Aghdam, Hongpeng Jin, and Yanzhao Wu. 2024. Da-moe: Towards dynamic expert allocation for mixture-of-experts models. *CoRR*, abs/2409.06669.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7432–7439. AAAI Press.

Krishna Teja Chitty-Venkata, Sandeep Madireddy, Murali Emani, and Venkatram Vishwanath. 2025. Lexi: Layer-adaptive active experts for efficient moe model inference. *Preprint*, arXiv:2509.02753.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *Preprint*, arXiv:1803.05457.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

DeepSeek-AI. 2024a. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *Preprint*, arXiv:2405.04434.

DeepSeek-AI. 2024b. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *J. Mach. Learn. Res.*, 23:120:1–120:39.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, and 1 others. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2024. The language model evaluation harness.

Yongxin Guo, Zhenglin Cheng, Xiaoying Tang, Zhaopeng Tu, and Tao Lin. 2025. Dynamic mixture of experts: An auto-tuning approach for efficient transformer models. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Quzhe Huang, Zhenwei An, Nan Zhuang, Mingxu Tao, Chen Zhang, Yang Jin, Kun Xu, Kun Xu, Liwei Chen, Songfang Huang, and Yansong Feng. 2024. Harder task needs more experts: Dynamic routing in MoE models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12883–12895, Bangkok, Thailand. Association for Computational Linguistics.

Wei Huang, Yue Liao, Jianhui Liu, Ruifei He, Haoru Tan, Shiming Zhang, Hongsheng Li, Si Liu, and Xiaojuan Qi. 2025. Mixture compressor for mixture-of-experts llms gains more. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, and 7 others. 2024. Mixtral of experts. *Preprint*, arXiv:2401.04088.

Harsh Kohli, Sachin Kumar, and Huan Sun. 2025. GroundCocoa: A benchmark for evaluating compositional & conditional reasoning in language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8280–8295, Albuquerque, New Mexico. Association for Computational Linguistics.

Harsha Kokel, Michael Katz, Kavitha Srinivas, and Shirin Sohrabi. 2025. Acpbench: Reasoning about action, change, and planning. In *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 26559–26568. AAAI Press.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2021. Gshard: Scaling giant models with conditional computation and automatic sharding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2021. Logiqa: a challenge dataset for machine reading comprehension with logical reasoning. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3622–3628.

Xudong Lu, Qi Liu, Yuhui Xu, Aojun Zhou, Siyuan Huang, Bo Zhang, Junchi Yan, and Hongsheng Li. 2024. Not all experts are equal: Efficient expert pruning and skipping for mixture-of-experts large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6159–6172, Bangkok, Thailand. Association for Computational Linguistics.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Shen-yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2020. A diverse corpus for evaluating and developing English math word problem solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 975–984, Online. Association for Computational Linguistics.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, Yuling Gu, Shane Arora, Akshita Bhagia, Dustin Schwenk, David Wadden, Alexander Wettig, Binyuan Hui, Tim Dettmers, Douwe Kiela, and 5 others. 2025. Olmoe: Open mixture-of-experts language models. *Preprint*, arXiv:2409.02060.

Alexandre Muzio, Alex Sun, and Churan He. 2024. Seer-moe: Sparse expert efficiency through regularization for mixture-of-experts. *CoRR*, abs/2404.05089.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, Berlin, Germany. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. 2023. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, Toronto, Canada. Association for Computational Linguistics.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, Zhuofu Chen,

Jialei Cui, Hao Ding, Mengnan Dong, Angang Du, Chenzhuang Du, Dikang Du, Yulun Du, Yu Fan, and 150 others. 2025. Kimi k2: Open agentic intelligence. *Preprint*, arXiv:2507.20534.

Qwen Team. 2024. Qwen1.5-moe: Matching 7b model performance with 1/3 activated parameters".

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 94–106, Copenhagen, Denmark. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Haoqi Yang, Luohe Shi, Qiwei Li, Zuchao Li, Ping Wang, Bo Du, Mengjia Shen, and Hai Zhao. 2025b. Faster moe llm inference for extremely large models. *Preprint*, arXiv:2505.03531.

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

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104, Brussels, Belgium. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Shuzhang Zhong, Ling Liang, Yuan Wang, Runsheng Wang, Ru Huang, and Meng Li. 2024. 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 2024, Newark Liberty International Airport Marriott, NJ, USA, October 27-31, 2024*, pages 51:1–51:9. ACM.

A Setup Details

A.1 Implementation Details of Baselines

Layer-level Baseline. The *Ascending* baseline distributes the budget B across L layers using a monotonically increasing schedule. It initializes an integer allocation via rounded linear interpolation between $K_{\min}$ and $K_{\max}$, producing a depth-increasing profile. This allocation is then refined through bi-directional passes to enforce monotonicity and constrain inter-layer differences to at most one. To exactly satisfy the global budget B, greedy adjustments are applied to internal layers while preserving monotonicity and boundary constraints. The *Descending* variant follows the same procedure in reverse order. In practice, we set $K_{\min} = 1$ and $K_{\max}$ to the model's original Top-K. When the budget B exceeds the feasible range under these bounds, $K_{\min}$ is increased until a valid allocation satisfying the budget constraint is obtained.

Token-level Baseline. *Dynamic-MoE* performs dynamic token routing by adjusting the number of expert activations per token based on a pre-profiled Top-P threshold. The threshold is calibrated under the *Uniform* layer-level allocation, ensuring that the resulting expert activations meet the target budget. *NAEE* implements Dynamic Expert Skipping by conditionally skipping experts based on routing scores. In our evaluation, we extend its original Top-2 routing to Top-K routing, skipping the k-th expert if its score falls below a layer- and model-specific threshold β relative to the Top-1 expert. This threshold is profiled under the *Uniform* layer-wise allocation to ensure the same target expert budget is maintained.

A.2 Efficiency Evaluation Details

For inference efficiency evaluation, we measure prefill and decode speedup ratios using dummy prompts with randomly generated tokens (batch size 8, prompt length 32, decode length 128). This configuration provides a controlled and reproducible environment that isolates the impact of expert allocation on inference latency. Speedups are averaged over 10 runs after a 5 warm-up iterations to reduce variability, and reported relative to the original configuration.

B Additional Results

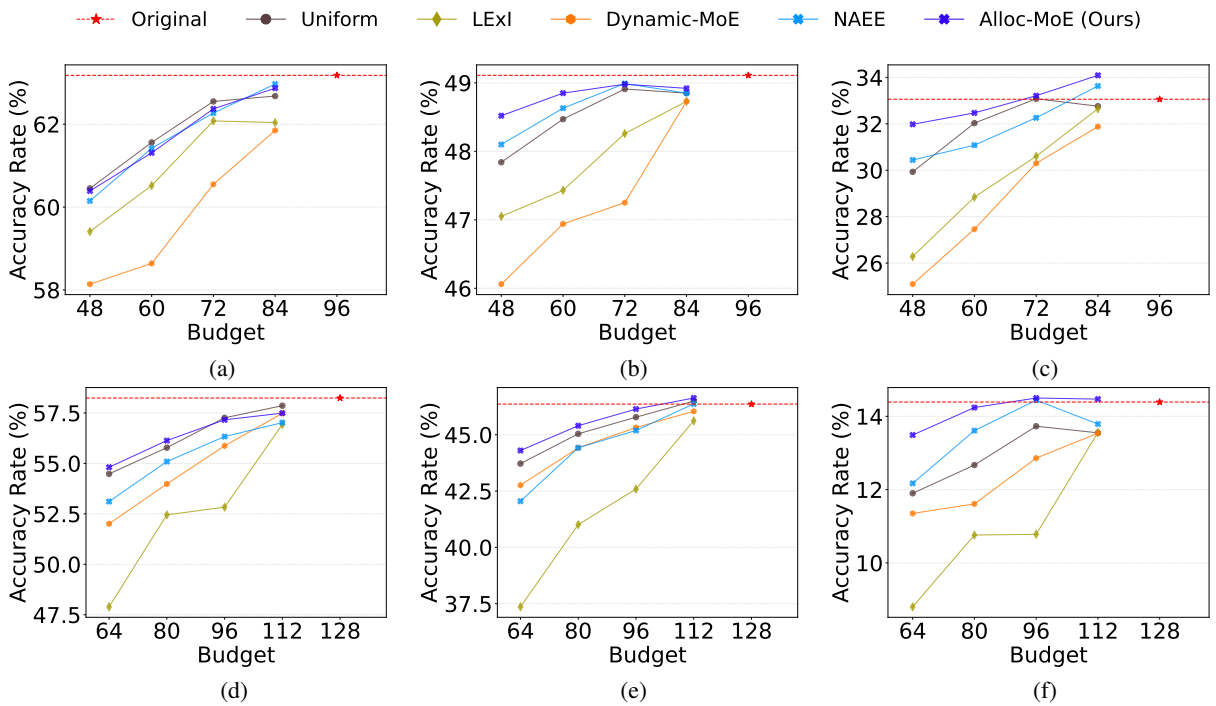


Figure 9: Alloc-MoE results on Qwen1.5-MoE-A2.7B and OLMoE-1B-7B-0924 under varying global activation budgets. (a–c) report results on Qwen1.5-MoE-A2.7B, while (d–f) correspond to OLMoE-1B-7B-0924. For each model, (a,d) show NLU tasks, (b,e) Reasoning tasks, and (c,f) Math tasks.

K_{base}	Budget				Avg.
	48	60	72	84	
0	46.87	47.86	48.06	48.63	47.86
1	**46.89**	47.84	48.23	48.66	**47.91**
2	46.00	**47.93**	**48.33**	**48.69**	47.74
3	46.00	47.90	48.25	48.48	47.66

Table 5: Analysis of K_{base} in Alloc-T on Qwen1.5-MoE-A2.7B under *Uniform* allocation strategy. Results are averaged across three task groups, with the last column showing the average performance across all budgets.

K_{base}	Budget				Avg.
	64	80	96	112	
0	37.27	38.47	39.14	39.55	38.61
1	37.38	38.50	39.09	**39.61**	38.65
2	37.39	38.49	**39.16**	39.55	38.65
3	**37.53**	**38.56**	39.05	39.51	**38.66**
4	36.77	**38.56**	39.04	39.49	38.47
5	36.77	37.97	38.96	39.61	38.33
6	36.77	37.97	39.11	39.54	38.35
7	36.77	37.97	39.11	39.47	38.33

Table 6: Analysis of the K_{base} in Alloc-T on OLMoE-1B-7B-0924 under *Uniform* allocation strategy. Results are averaged across three task groups, with the last column showing the average performance across all budgets.

Method	Budget				Avg.
	48	60	72	84	
Uniform	46.07	47.35	48.18	48.10	47.43
+L	46.11	47.78	**48.33**	48.59	47.70
+T	46.89	**47.84**	48.19	**48.66**	**47.90**
+L +T	**46.96**	47.54	48.19	48.63	47.83

Table 7: Ablation study of Alloc-L (**+L**) and Alloc-T (**+T**) on Qwen1.5-MoE-A2.7B under varying global activation budgets. Results are averaged across three task groups, with the last column showing the average performance across all budgets.

Method	Budget				Avg.
	64	80	96	112	
Uniform	36.70	37.83	38.93	39.30	38.19
+L	37.08	38.21	39.09	39.35	38.43
+T	37.38	38.50	39.06	**39.61**	38.64
+L +T	**37.53**	**38.59**	**39.27**	39.53	**38.73**

Table 8: Ablation study of Alloc-L (**+L**) and Alloc-T (**+T**) on OLMoE-1B-7B-0924 under varying global activation budgets. Results are averaged across three task groups, with the last column showing the average performance across all budgets.

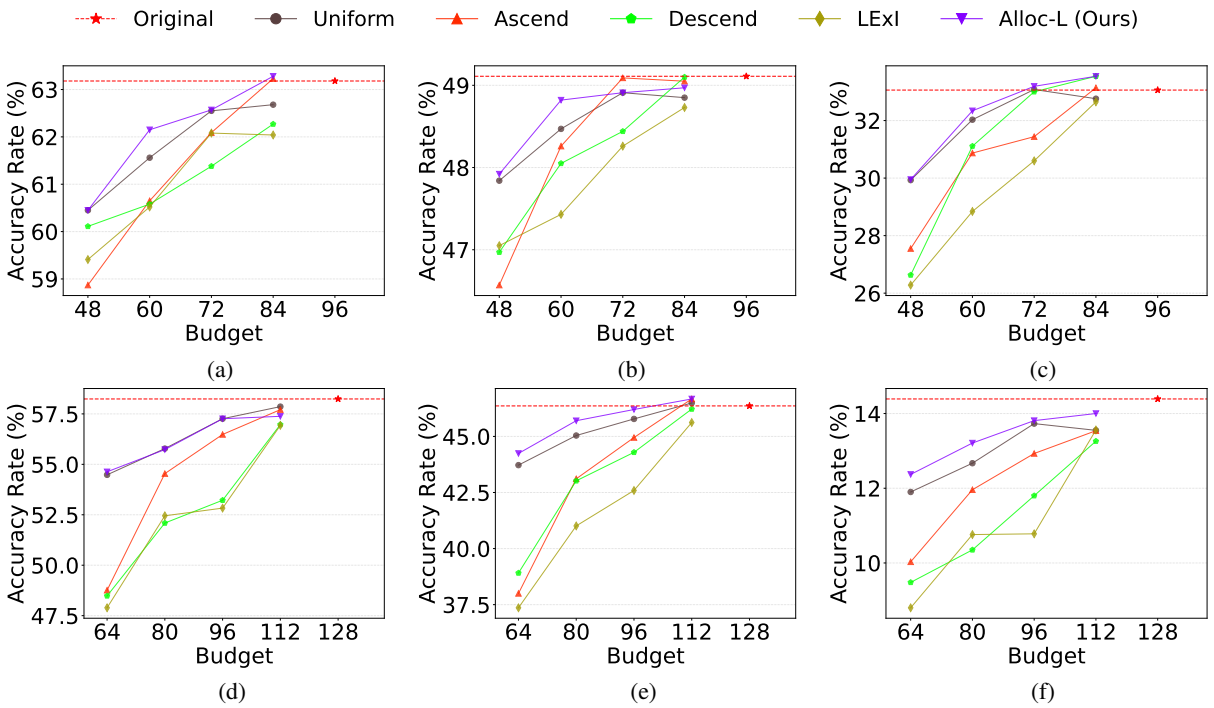


Figure 10: Ablation results of Alloc-L on Qwen1.5-MoE-A2.7B and OLMoE-1B-7B-0924 under varying global activation budgets. (a–c) report results on Qwen1.5-MoE-A2.7B, while (d–f) correspond to OLMoE-1B-7B-0924. For each model, (a,d) show NLU tasks, (b,e) Reasoning tasks, and (c,f) Math tasks.

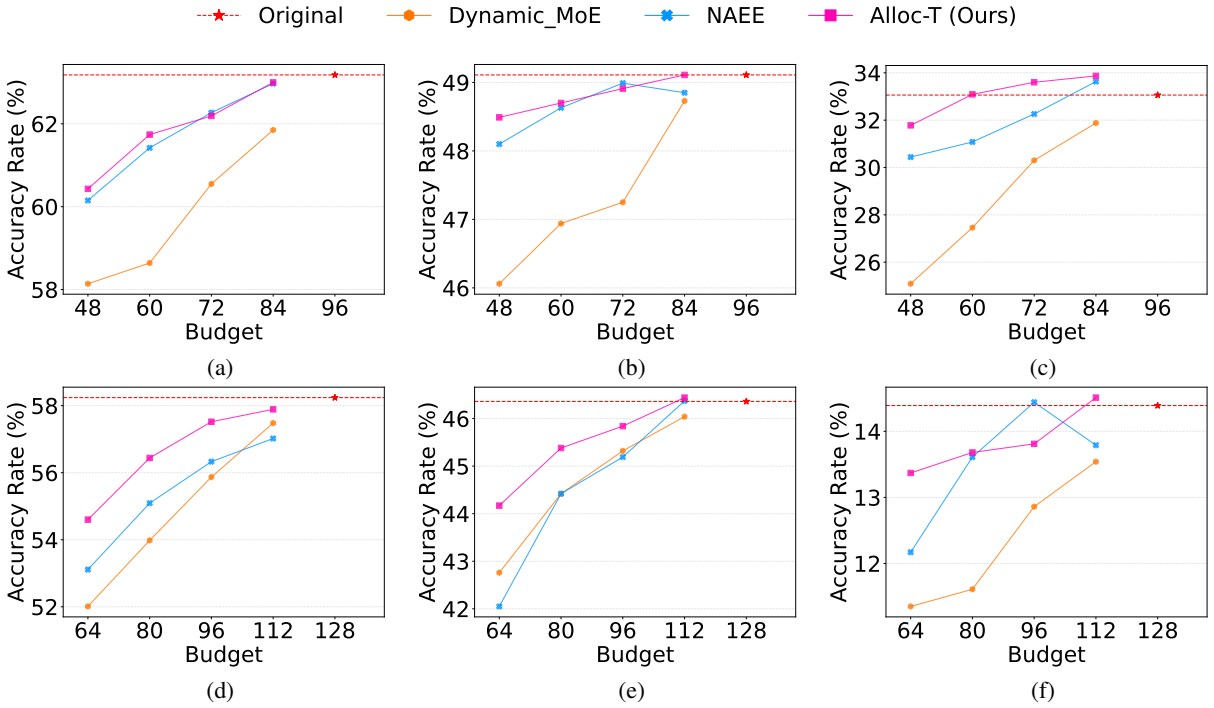
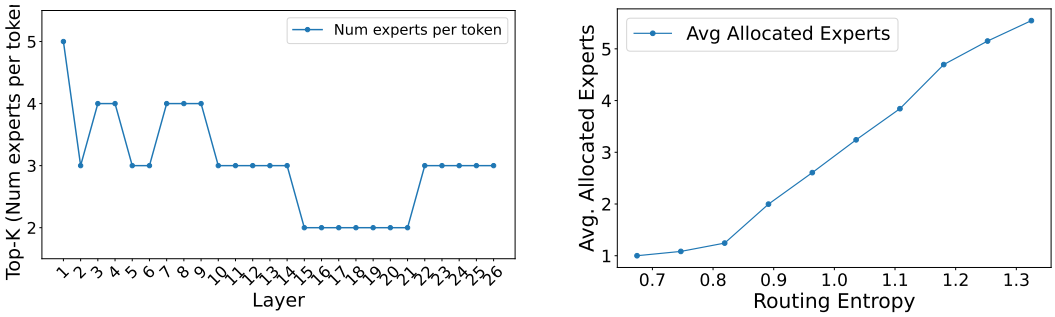


Figure 11: Ablation results of Alloc-T on Qwen1.5-MoE-A2.7B and OLMoE-1B-7B-0924 under varying global activation budgets. (a–c) report results on Qwen1.5-MoE-A2.7B, while (d–f) correspond to OLMoE-1B-7B-0924. For each model, (a,d) show NLU tasks, (b,e) Reasoning tasks, and (c,f) Math tasks.



(a) (b)

Figure 12: (a) Layer-wise allocation of Alloc-MoE on DeepSeek-V2-Lite with Budget = 78, demonstrating a clearly non-uniform distribution across layers. Earlier layers retain larger expert budgets, while deeper layers operate with fewer activated experts. (b) Entropy results on the 10-th MoE layer of DeepSeek-V2-Lite. Low-entropy (high-confidence) tokens retain fewer experts, whereas high-entropy (ambiguous) tokens retain more. Similar patterns are observed across layers.