Bag of Tricks for Sparse Mixture-of-Experts: A Benchmark Across Reasoning, Efficiency, and Safety

Mufan Qiu*[1], Zheyu Shen*[2], Pingzhi Li[1]
Ang Li†[2], Tianlong Chen†[1]

[1]University of North Carolina at Chapel Hill [2]University of Maryland
*Equal Contribution †Equal Supervision

Abstract

Mixture-of-Experts (MoE) has emerged as a promising approach for scaling large language models efficiently. However, *how to design a desired MoE architecture* given performance, efficiency, or safety goals remains absent. Existing benchmarks often focus on isolated aspects (e.g., reasoning, efficiency, safety), and there is a lack of consensus on optimal design choices, such as the number and size of experts, the type of routers, and the regularization during pre-training, or strategies like freezing, learning rate adjustments, and limiting expert collaboration during fine-tuning, with prior works often yielding conflicting conclusions. Motivated by this research gap, we introduce `MoE-Bench`, the first comprehensive assessment of MoE designs across the three dimensions of *reasoning ability*, *efficiency*, and *safety*. Our benchmark systematically evaluates optimal architectural choices during both pre-training and fine-tuning phases. We evaluate two popular MoE backbones across four dimensions of design choices on over eight metrics. Our empirical findings uncover hidden underlying correlations among MoE design choices. Specifically, we observe that (1) token-level routing and z-loss regularization improve reasoning performance; (2) shared experts enhance training stability but reduce specialization; and (3) collaboration-constrained routing and freezing strategies significantly influence load balance, specialization, and safety alignment. Furthermore, we propose three "sweet point" combinations of optimal strategies tailored to different scenarios. We hope this study provides actionable insights for building more robust, efficient, and secure MoE models. Code, checkpoints, and raw data will be released upon acceptance of the paper.

1 Introduction

The pursuit of enhanced capabilities in Large Language Models (LLMs) has demonstrated a strong correlation between model scale and performance (Achiam et al., 2023; Yang et al., 2024; Kaplan et al., 2020). However, naively increasing model size leads to prohibitive computational costs for training and substantial inference latency (Kaplan et al., 2020). To mitigate these challenges, various techniques have been investigated, including quantization (Lin et al., 2024; Frantar et al., 2022), low-precision training (Wang et al., 2023), and sparse computing (Shazeer et al., 2017; Lepikhin et al., 2020; Frankle and Carbin, 2018). Among these diverse approaches, Mixture of Experts (MoE) has emerged as a particularly compelling paradigm, adopted by numerous state-of-the-art (SoTA) models (Grattafiori et al., 2024; Guo et al., 2025; Achiam et al., 2023). The fundamental principle of MoE architectures involves selectively activating only a sparse subset of the model's parameters for each input token, thereby decoupling the total parameter count from the per-token computational load. MoE first demonstrated significant success in language modeling (Shazeer et al., 2017) and machine translation (Lepikhin et al., 2020), and its use has since been extended to computer vision (Riquelme et al., 2021) and multi-modal applications (Mustafa et al., 2022). A primary driver of MoE's success lies in its capacity to expand model size cost-effectively while maintaining favorable scaling properties (Krajewski et al., 2024). Moreover, the inherent sparse activation mechanism within MoE models offers potential avenues for enhanced model interpretability (Lewis et al., 2021; Zoph, 2022).

Despite the remarkable success of MoE models, their intricate design introduces significant complexities. The choice of expert architecture and routing mechanisms, for instance, profoundly impacts the model's reasoning capabilities (Krajewski et al., 2024; Zhou et al., 2022). Furthermore, considerations such as inter-expert load balancing and collaborative behavior necessitate the introduction of auxiliary regularization techniques to

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 817–835

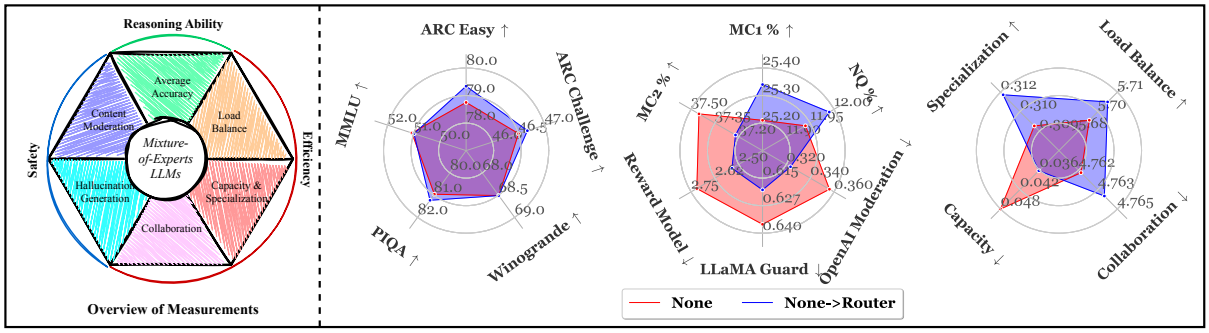

Figure 1: Overview of `MoE-Bench`. Left: Evaluation of various MoE designs during pretraining and fine-tuning across Reasoning Ability, Safety, and Efficiency dimensions using over six metrics, including analysis of their intercorrelations. Right: "Sweet point" fine-tuning techniques demonstrate comprehensive improvements over the baseline in performance, safety, and efficiency. `None` denotes full-parameter fine-tuning; `None->Router` denotes full-parameter fine-tuning for the initial training phase, then fine-tuning with a frozen router.

maintain efficiency and performance (Zoph et al., 2022; Wang et al., 2024; Zhang et al., 2025). Compared to their dense counterparts, MoE models offer additional avenues for optimization, such as expert pruning (Li et al., 2023) and enhancing inter-expert communication protocols (Zhang et al., 2025), which can further reduce inference latency but also add layers to the design complexity. However, existing research on MoE models predominantly focuses on singular evaluation dimensions, such as reasoning ability, often neglecting a holistic assessment across multiple critical aspects including efficiency and safety (Fu et al., 2024; Chen et al., 2024). Critically, prevalent metrics for evaluating load balancing are often inadequate for assessing the efficacy of many contemporary MoE optimization strategies utilizing expert activation or collaboration(Zhang et al., 2025; Luo et al., 2025). This yields the primary question to be explored:

> ***(Q)*** *Can we establish a comprehensive benchmarking framework combined with a bag of tricks, to significantly enhance the design, efficiency, and safety of MoE models across diverse tasks and architectures?*

Motivated by these limitations and the research question, we introduce **`MoE-Bench`**, a comprehensive benchmark designed to rigorously evaluate MoE architectures across multiple performance dimensions. As illustrated in Figure 1, **`MoE-Bench`** quantifies and assesses a suite of metrics spanning three key domains: (1) We introduce novel concepts of expert *capacity* and *specialization* to gauge the importance and potential redundancy of individual experts. (2) We incorporate metrics for evaluating expert collaboration, offering insights for the development of advanced optimization techniques. (3) We conduct a thorough assessment of model-generated content for hallucinations and safety. Furthermore, **`MoE-Bench`** systematically explores the vast MoE design space, scrutinizing a range of MoE-specific hyperparameters, including expert granularity, load balancing schemes, and routing strategies, to identify optimal design configurations.

Our contributions are summarized as follows:

- **First Comprehensive Evaluation of MoE Training Across Multiple Dimensions:** Going beyond prior studies that primarily focus on isolated aspects such as reasoning ability or training cost (Fu et al., 2024; Cai et al., 2025), we present the first systematic and unified evaluation of MoE training techniques across three key dimensions: reasoning capability, reliability and general utility. Our analysis reveals several key insights: ❶ The fine-tuning strategy of full-parameter fine-tuning followed by freezing the router yields the best reasoning performance and safety alignment. Simultaneously, its load balance, capacity, and collaboration also enable better application of optimization techniques like pruning and Collaboration-Constrained Routing (C2R) to enhance efficiency. ❷ We observed inherent correlations among MoE model efficiency metrics such as load balance, expert collaboration, and specialization. This overall perspective enables a more balanced and in-depth understanding of what truly drives performance in MoE models.

- **Unified and Reproducible Evaluation**

Framework: We propose a standardized evaluation setup with consistent hyperparameters, dataset splits, and training protocols, implemented within a modular and extensible codebase. To achieve this, we integrated all MoE training strategies into a common infrastructure built on top of OLMoE (Muennighoff et al., 2024) and MegaBlocks (Gale et al., 2023) libraries, ensuring that each experiment runs under identical conditions. This unified framework enables fair and reproducible comparisons across different MoE training strategies. This design novelty isolates the effect of routing and expert selection mechanisms by controlling for all other variables, enabling truly fair comparisons. Furthermore, all configurations and scripts are released publicly, supporting full reproducibility and extensibility by the community.

➢ **Bag of Tricks for MoE Training Technique Combinations:** We investigate the interactions between various MoE training methods across OLMoE and DeepSeek MoE backbones and over six datasets such as Massive Multitask Language Understanding (MMLU), Winogrande, and Hellaswag, and also includes three safety datasets such as Malicious, CoNa, and Controversial. We find that our identified sweet-point fine-tuning strategy of full-parameter fine-tuning followed by freezing the router achieves significant improvements over the baseline across all three dimensions: reasoning, efficiency, and safety. This finding offers practical guidelines for selecting and integrating MoE strategies targeting diverse performance requirements.

2 Related Works

2.1 Basic Architecture of Mixture of Experts

A MoE layer replaces a standard feed-forward network within a transformer block. Given an input token representation x, the output of an MoE module, y, is a weighted sum of the outputs from its n expert networks $\{E_0, \ldots, E_{n-1}\}$, as $y = \sum_{i=0}^{n-1} G(x)_i \cdot E_i(x)$, where $E_i(x)$ is the output of the i-th expert network when processing input x. The term $G(x)_i$ represents the gating weight assigned by the router network $G(\cdot)$ to the i-th expert for input x. The design of the router network $G(\cdot)$ is a critical component and can vary across different MoEs (Fedus et al., 2021; Lepikhin et al., 2020).

A prevalent routing strategy is the top-k gating mechanism. In this approach, the router, typically a linear layer followed by a softmax function, calculates a probability distribution over the experts. The top k experts with the highest probabilities are selected to process the input. The gating values $G(x)$ are formally computed as:

$$G(x) = \texttt{Top-K}(\text{softmax}(W_g x)) \quad (1)$$

where W_g is the learnable weight matrix of the router's linear layer. For fine-grained control over expert selection and to prevent representational collapse or expert starvation, MoE models are often trained with auxiliary loss functions. During pretraining, a load balancing loss is commonly employed to encourage a more uniform distribution of tokens across experts (Lepikhin et al., 2020; Shazeer et al., 2017; Zoph et al., 2022). Conversely, some fine-tuning strategies might employ losses that encourage specialization, concentrating assignments to a few experts for specific downstream tasks (Zhang et al., 2025).

2.2 Pretrain Designs

The pretrain phase is pivotal for establishing the foundational capabilities of MoE models. Key design decisions during this phase, which significantly influence subsequent model characteristics, revolve around three main areas: expert architecture, router design, and regularization strategies. Our exploration covers variations in expert granularity and the inclusion of shared experts (Dai et al., 2024); different router choice strategies and granularities (Shazeer et al., 2017; Fedus et al., 2021; dos Santos et al., 2023); and diverse regularization techniques including auxiliary losses and auxiliary loss-free approaches (Shazeer et al., 2017; Lepikhin et al., 2020; Zoph et al., 2022). For a detailed exposition of these design considerations and the specific combinations investigated, please refer to Appendix B.1.

2.3 Finetuning Designs

While MoE models benefit from large-scale pretraining, their unique structure introduces specific considerations during finetuning. Unconstrained updates can potentially destabilize learned routing patterns or degrade expert specialization.

2.3.1 Freeze Strategy: Freeze Router/Expert & Different Learning Rates

Selectively freezing parts of the MoE architecture is one approach to stabilize finetuning. This can involve: ❶ **Freezing the Router** (Dai et al., 2022): This strategy aims to maintain the expert assignment patterns learned during pretraining. This can be particularly relevant when finetuning on domain-shifted data where re-learning routing might be challenging. ❷ **Freezing Experts** (Seo et al., 2025): This approach preserves the functional specialization of experts acquired during pretraining, which can be beneficial when downstream task data is limited. ❸ **Interleave Freezing** (Dai et al., 2022): This involves a combination, such as freezing the router while updating experts, or vice-versa, potentially in different phases of finetuning.

In conjunction with freezing strategies, employing different learning rates for different components, such as a smaller learning rate for the router (Dikkala et al., 2023), can also be used. This allows experts to adapt more readily to downstream tasks while preserving the more slowly evolving routing patterns.

3 Bag of Tricks for Improving MoE Reasoning Ability

3.1 Experiments Setup

We evaluate two representative MoE models (OLMoE (Muennighoff et al., 2024) and DeepSeek MoE (Guo et al., 2025)) across pre-training and fine-tuning stages using Minipile (Kaddour, 2023a) and LIMA datasets (Zhou et al., 2023). These models differ significantly in scale, routing strategies, and architectural design. Full model and dataset details are provided in Appendix B.2 and Appendix B.3.

3.2 Evaluation Metrics

To comprehensively evaluate the reasoning ability of MOE models, we use a range of reasoning and knowledge-intensive tasks: MMLU, ARC-easy, ARC-challenge, Physical Interaction QA (PIQA), Winogrande, and Hellaswag. For the detail of these matrices, please refer to Appendix B.5.

3.3 Takeaways

3.3.1 Impact of Pretraining Expert Architecture on Reasoning Performance

Contrasting Shared Expert Performance in Different Architectures. Our analysis of the relationship between model reasoning ability and architectural design reveals several interesting and counter-intuitive results. For instance, under identical training datasets and evaluation setups, shared experts in the DeepSeek MoE architecture demonstrate consistent improvements in reasoning capabilities over dense models. However, the shared expert design in OLMoE does not exhibit a similar advantage, as illustrated in Figure 2. This consistent discrepancy, aligning with the respective findings of their original papers, suggests that specific architectural differences may significantly influence the efficacy of shared experts.

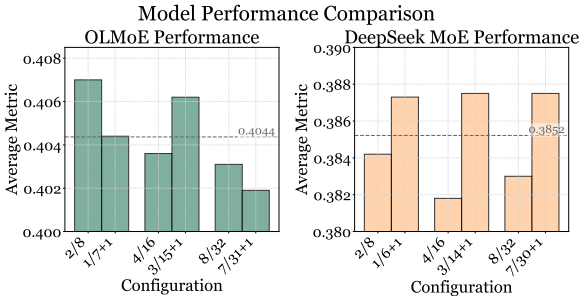

Figure 2: Reasoning ability comparison of OLMoE and DeepSeek MoE with different expert granularity and shared expert settings. The dashed line indicates the average performance across six settings. $3/15 + 1$ denotes a configuration with 3 top experts selected from 15 routed experts, augmented by one fixed, always-activated shared expert.

Suboptimal Efficacy of Fine-Grained Experts. Furthermore, in our experiments, fine-grained experts did not perform optimally. As seen in Figure 3, fine-grained experts also showed no clear advantage in terms of training convergence speed. This contradicts the conclusions from both OLMoE and DeepSeek MoE studies. We hypothesize that model size and vocabulary size might be contributing factors to this disparity, warranting further investigation. There remains considerable scope for exploring the underlying principles of existing MoE designs.

3.3.2 Impact of Pretraining Regularization on Model Convergence and Performance

Coarse-Grained Experts Consistently Achieve Faster Convergence. Figure 3 illustrates the training loss curves for models during pretraining. We observe that: Across all four settings, coarse-grained experts achieve faster convergence speeds compared to fine-grained experts. The presence of shared experts, the removal of auxiliary loss (Aux

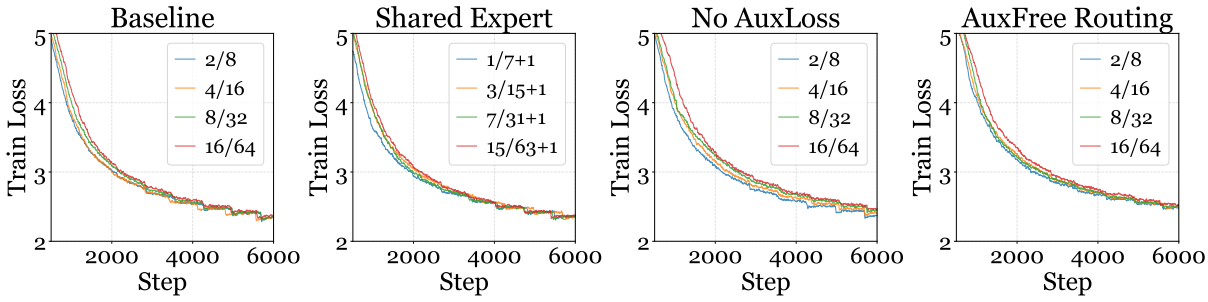

Figure 3: OLMoE pretraining loss curves under different design configurations. AuxFree Routing refers to using bias for load balancing(Wang et al., 2024). The Baseline setting indicates no shared expert, presence of auxiliary loss, and no bias for load balancing.

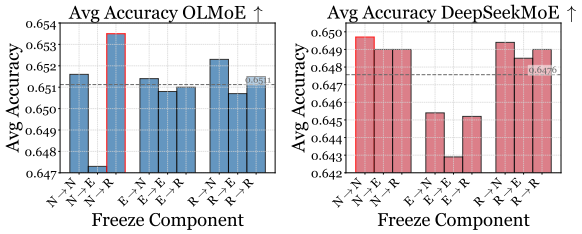

Figure 4: Average accuracy of OLMoE and DeepSeek MoE under different fine-tuning freezing strategies. N→E indicates no freezing in the first half of training and freezing the router in the second half. The average performance across nine settings is marked with a dashed line.

Free Routing), or the use of bias for load balancing did not alter this trend.

Auxiliary Loss Absence Amplifies Convergence Disparities. Compared to the baseline setting, the "aux-free" setting exhibits a more pronounced difference in convergence speeds between different granularities of experts. This could be because the auxiliary loss in baseline settings typically aids in stabilizing the training of fine-grained experts by promoting more balanced expert utilization; its absence in "aux-free" settings may therefore accentuate any inherent instabilities or slower learning characteristics of fine-grained experts. For more detailed performance metrics of each setting, please refer to Appendix C.1.

3.3.3 Influence of Fine-tuning Freezing Strategies on Reasoning Capabilities

Early-Stage Freezing Impacts Reasoning More. Our experiments reveal that:The freezing strategy employed during the first half of fine-tuning has a more pronounced impact on the model's reasoning ability compared to strategies applied in the latter half. We have grouped our experimental results accordingly. This phenomenon might be attributed to the learning rate scheduler.

Expert Freezing Detrimentally Affects Reasoning Ability. Freezing experts during the second half of training consistently results in inferior performance compared to the other two freezing strategies (freezing the router or no freezing). Furthermore, freezing experts during the first half of training for DeepSeek MoE also led to a significant performance degradation. This underscores the necessity of keeping experts unfrozen during fine-tuning to maintain reasoning capabilities.

Full Fine-tuning and Router Freezing Yield Comparable Performance. Full parameter fine-tuning and freezing only the router show comparable performance, suggesting flexibility in choosing between these two approaches when expert parameters are active.

3.4 Summary of Reasoning Ability Findings

Our investigation into MoE models' reasoning abilities yields several key insights. During the pretraining phase, the design choices for shared experts and the granularity of experts (coarse vs. fine) exert a consistent influence on reasoning performance. However, the optimal design appears to be contingent upon the specifics of the backbone architecture. Regularization losses, such as auxiliary load balancing losses, can help mitigate some of the reasoning performance discrepancies arising from these architectural choices. In the fine-tuning stage, the freezing strategy adopted in the initial phase is more critical than in later stages. Notably, freezing expert parameters consistently leads to suboptimal reasoning performance, highlighting the importance of allowing experts to adapt during fine-tuning.

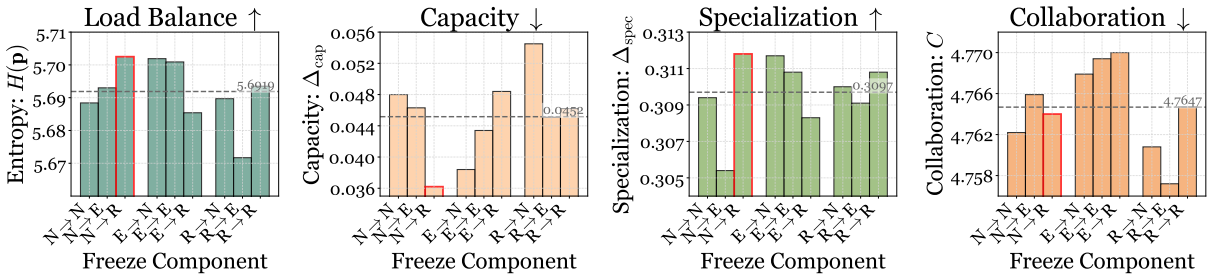

Figure 5: Finetuning Capacity Analysis. This figure illustrates the relationship between different freezing settings and model efficiency metrics. N denotes freezing None, E denotes freezing Experts, and R denotes freezing the Router. X→Y (where X, Y ∈ {N, E, R}) indicates freezing component X in the first half of fine-tuning and component Y in the second half. The results are grouped by the freezing setting in the first stage. We observe that the freezing setting in the initial phase has a more significant impact on model's efficiency metrics, potentially due to the learning rate scheduler. The dashed line represents the average efficiency for each metric across all nine settings. We have highlighted the border of the sweet point setting (N→R).

4 Bag of Tricks for Improving MoE Efficiency

4.1 Evaluation Metrics

To systematically evaluate the efficiency of MoE models, we adopt four key metrics, each reflecting a distinct aspect of expert utilization:

Load Balance. We measure the uniformity of expert selection using the entropy of the routing distribution $H(\mathbf{p}) = -\sum_{i=1}^{N} p_i \log p_i$, where p_i denotes the probability of selecting expert i among N experts. A higher entropy indicates a more balanced load across experts, which is desirable for efficient resource utilization(Fedus et al., 2021; Zoph et al., 2022).

Capacity. To assess whether all experts are effectively utilized and not collapsed during training, we define capacity as the performance gap between the full model and a pruned variant where the least activated expert is removed for each MoE layer $\Delta_{\text{cap}} = \text{ACC}_{\text{Base}} - \text{ACC}_{\text{Pruned}}$, where ACC_{Base} is the accuracy of the original model and $\text{ACC}_{\text{Pruned}}$ is the accuracy after pruning the least utilized expert. A smaller gap indicates better capacity utilization.

Specialization. We quantify the degree to which experts learn distinct functions by comparing the model's performance with normal routing *v.s.* under random routing: $\Delta_{\text{spec}} = \text{ACC}_{\text{Base}} - \text{ACC}_{\text{Random}}$, where $\text{ACC}_{\text{Random}}$ is the accuracy when expert assignments are randomized. A larger gap suggests higher specialization among experts.

Collaboration. To evaluate the tendency of experts to collaborate (i.e., be co-activated by the same tokens), we compute the co-activation entropy for each expert $C_i = -\sum_{\substack{j=1 \\ j\neq i}}^{N} q_{ij} \log q_{ij}$, where q_{ij} denotes the empirical probability that experts i and j are co-activated. A higher C_i indicates that expert i is more likely to collaborate with others. Experts that frequently collaborate can be preferentially placed on the same device, thereby reducing communication overhead in distributed training and improving overall efficiency(Zhang et al., 2025; Luo et al., 2025).

4.2 Takeaways

Our analysis of efficiency metrics, particularly during the fine-tuning of OLMoE models, reveals several key insights and trade-offs. We observe a general positive correlation between Load Balance, Specialization, and Collaboration, whereas Load Balance tends to exhibit a negative correlation with Capacity. This holds throughout the training process 6. This section primarily discusses the impact of various metrics on model efficiency during fine-tuning; for the influence of different designs on model efficiency metrics during the pre-training phase, please refer to Appendix D.1.

4.2.1 Trade-offs and Interdependencies Among Efficiency Metrics

Metrics Exhibit Dynamic Trends and Interrelations During Fine-tuning. ❶ Specifically, as illustrated in Figure 5, which tracks these metrics over fine-tuning epochs, Load Balance, Specialization, and Collaboration typically reach their peak values in the early stages of fine-tuning before subsequently declining. ❷ In stark contrast, Capacity generally demonstrates a continuous increase, presenting an inverse trend to the other three metrics. This suggests that while initial fine-tuning may rapidly optimize for expert differentiation and utilization balance, prolonged training tends to enhance overall expert capacity, potentially at the

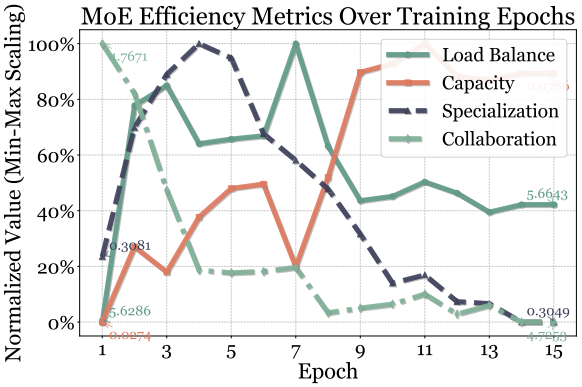

Figure 6: Efficiency Metrics Overview. This figure shows the changes in four efficiency metrics during the full parameter fine-tuning of OLMoE using LIMA, as training epochs progress. For ease of comparison, all metrics are normalized. We have marked the specific values of each metric at the first and last epochs on the graph.

expense of these initial optima. ❸ Furthermore, as depicted in Figure 6, experiments involving freezing different model components during fine-tuning consistently show a positive interrelation among Load Balance, Specialization, and Collaboration, suggesting that interventions targeting one can beneficially impact the others under such conditions. For additional experimental results, please refer to Appendix A.2.

Load Balance Decline Suggests Router Over-training. The observed trajectory of Load Balance—an initial increase followed by a decline—aligns with prior research identifying router over-training as a potential issue (Dikkala et al., 2023). This implies that strategic interventions, such as freezing the router or reducing its learning rate in later training phases, could potentially sustain or improve load balance.

Consistent Capacity Growth Favors Early Stopping for Prunability. Concurrently, the consistent growth of Capacity throughout training suggests an increasing importance of experts that were initially less frequently activated. This observation indicates that early stopping might be an effective strategy to maintain model pruning performance, as it captures a state where expert roles are less entrenched and low-activity experts are more readily dispensable.

Diminishing Collaboration Aligns with Reduced Load Balance, Aiding Restriction Strategies. Finally, Collaboration is observed to generally diminish throughout training. This decline is potentially linked to the diminishing load balance observed in later epochs. This evolution creates an opportunity for efficiency optimization that communication overhead can be restricted in later training stages with minimal performance impact, as the model organically develops reduced expert collaboration (Zhang et al., 2025; Luo et al., 2025).

4.2.2 Impact of Freezing Strategies on Model Efficiency Metrics

Our experiments with different freezing strategies during fine-tuning reveal significant implications for MoE efficiency metrics. As evidenced in Figure 5 demonstrates that different freezing strategies produce distinct patterns across efficiency metrics, illuminating critical trade-offs in the optimization of MoE architectures.

Freezing Experts Decreases Capacity and Increases Collaboration. When experts are frozen during fine-tuning, we observe a notable decrease in Capacity measurements. This indicates that pruning the least active expert has a diminished performance impact. Simultaneously, the router compensates for the static expert parameters by facilitating enhanced inter-expert collaboration, leading to a substantial increase in Collaboration. This demonstrates the adaptive nature of the routing mechanism freezing confronted wit in the expert modules.

Freezing Router Heightens Capacity but Complicates Pruning. Conversely, when the router is frozen, each expert maintains its established specialization while continuing to be fine-tuned within that designated role. This combination preserves the functional differentiation among experts while allowing for parameter refinement, resulting in heightened Capacity measurements. Consequently, pruning even the least frequently activated expert may significantly degrade model performance, as each expert has been optimized for its specific function domain. This finding suggests that router freezing strategies, while beneficial for maintaining load balance as previously observed, may complicate expert pruning efforts in scenarios where model compression is desired.

Early Router Freezing Reduces Load Balance and Collaboration, Aiding Inference Optimization. Furthermore, as shown in Figure 5, freezing the router in the first half of fine-tuning (e.g., R→N, R→E, R→R strategies) tends to decrease Load Balance but also results in lower Collaboration. This lower collaboration can be advantageous for applying optimizations that reduce inter-expert communication, thereby enhancing model inference

efficiency(Zhang et al., 2025; Luo et al., 2025).

Validation on Larger Scale Model. To validate our findings at scale, we conducted experiments on the 16B DeepSeek-MoE model. As illustrated in Figure 7, our proposed fine-tuning strategy (N→R) achieved the strongest collaboration score, second-best load balance and specialization, and third-highest average accuracy among all tested configurations. These results affirm that our conclusions generalize to large-scale settings. Furthermore, we are collaborating with the OLMoE team to evaluate additional pre-training checkpoints that vary in expert granularity, shared expert design, and auxiliary loss usage. While this extended validation is ongoing, we commit to including a comprehensive large-scale analysis in the final version to strengthen cross-validation of our findings.

Freezing Strategies Highlight Specialization-Redundancy Trade-off. These observations underscore the nuanced relationship between freezing strategies and efficiency metrics, revealing that architectural decisions during fine-tuning must carefully consider the downstream deployment constraints and optimization objectives. The trade-off between expert specialization and redundancy emerges as a critical consideration when implementing freezing strategies in MoE optimization.

4.2.3 Impact of Router Learning Rate on Model Efficiency Metrics

Motivation for Differential Router Learning Rates. Our preceding analysis, along with prior research, indicates a decline in Load Balance as training progresses, potentially attributable to router over-training (Dikkala et al., 2023). Observations from Figure 5 show that freezing the router during the latter half of fine-tuning yields optimal Load Balance. Conversely, freezing the router in the initial half results in Load Balance figures that are merely average or sub-par compared to other fine-tuning configurations. This discrepancy prompts further investigation into the effects of applying a consistently smaller learning rate to the router component throughout the training duration, as an alternative to outright freezing, to ascertain its impact on load balancing.

Reduced Router Learning Rate Diminishes Load Balance. Contrary to expectations, the experimental results presented in Figure 8 demonstrate that employing a smaller learning rate for the router does not enhance Load Balance. In fact, for both DeepSeek MoE and OLMoE models, Load Balance deteriorates as the router's learning rate is reduced. This outcome suggests that implementing an early stopping strategy specifically for the router might prove to be a more efficacious approach for maintaining or improving load distribution.

4.3 Identifying a Sweet Point for Efficiency-Focused Fine-tuning

Optimal Strategy: Unrestricted Initial Phase Followed by Router Freezing. For fine-tuning paradigms prioritizing efficiency, a particularly compelling strategy emerges: maintaining an unrestricted training regimen (i.e., no components frozen) during the initial phase, followed by freezing the router in the subsequent phase (denoted as N→R). Our empirical evaluations on OLMoE reveal that this approach consistently achieves superior outcomes across multiple efficiency dimensions when juxtaposed with alternative fine-tuning strategies. Specifically, the N→R configuration yields the most favorable Load Balance, minimizes the performance degradation post-pruning (Capacity), and fosters the highest degree of expert Specialization. Furthermore, it maintains a Collaboration level below the observed average, a characteristic conducive to optimizations aimed at reducing inter-expert communication overhead. Notably, analogous tendencies are discernible in experiments conducted with DeepSeek MoE as mentioned in Appendix C.2, reinforcing the proposition that freezing the router in the latter stages of fine-tuning represents a "sweet point" for optimizing MoE model efficiency.

5 Bag of Tricks for Improving MoE Safety

Due to space constraints, please refer to Appendix A for the discussion on safety.

6 Conclusion

In this work, we present **MoE-Bench**, the first comprehensive benchmark for evaluating MoE architectures across reasoning, efficiency, and safety. Our findings reveal key trade-offs and highlight N→R (no freezing then router freezing) as a particularly effective fine-tuning strategy, balancing resource efficiency and model performance. These results emphasize the importance of co-designing pretraining and fine-tuning to build MoE models that are not only powerful but also safe and efficient.

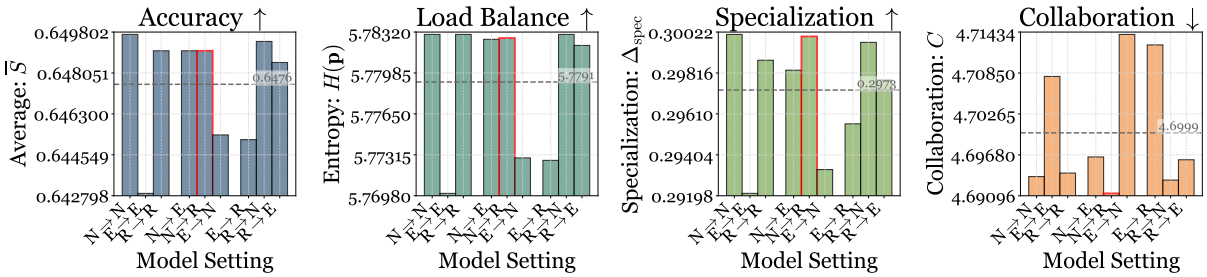

Figure 7: Performance of DeepSeek-MoE-16B under different fine-tuning strategies. This figure presents four key metrics: average accuracy, Load Balance, Specialization, and Collaboration. The N→R strategy (full-parameter training followed by router freezing) demonstrates strong performance, achieving the best collaboration score, second-best load balance and specialization, and third-highest average accuracy, validating its effectiveness on a large-scale model.

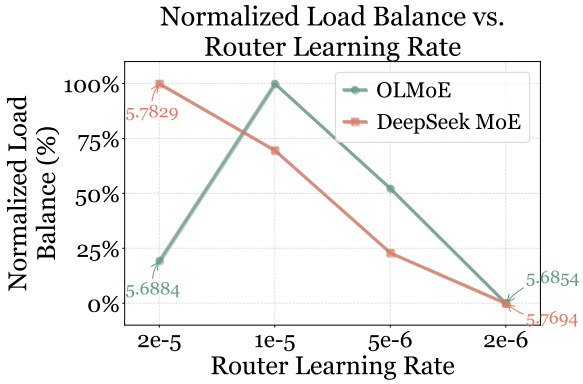

Figure 8: Relationship between router learning rate and Load Balance during fine-tuning. Load Balance values are normalized for comparison.

MoE-Bench offers a solid foundation for advancing responsible MoE development.

We acknowledge that our work does not propose a new algorithmic method, but rather centers on benchmarking. Our primary contribution lies in designing and executing a systematic and reproducible evaluation that unifies the critical axes of reasoning, efficiency, and safety. By consolidating these dimensions into a unified framework, accompanied by an open codebase, we surface hidden trade-offs and provide a robust foundation for future MoE research. We believe this fills an important gap in the literature and enables more informed and targeted development of MoE architectures.

Limitation

While our benchmark provides a comprehensive evaluation of MoE design choices, it has several limitations. First, the space of MoE architectures is vast, and we only cover a subset of commonly used expert routing and training strategies. Second, due to resource constraints, our experiments are limited to small-scale MoE models, which may not fully reflect the dynamics observed at larger scales. Third, our study focuses solely on language-only models, while ~~we leave the extension to multimodal MoE architectures for future work.~~we designed MoE-Bench to support extensibility and will provide guidance for adaptation to multimodal MoE architectures.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. 2025. A survey on mixture of experts in large language models. *IEEE Transactions on Knowledge and Data Engineering*.

Guanjie Chen, Xinyu Zhao, Tianlong Chen, and Yu Cheng. 2024. Moe-rbench: Towards building reliable language models with sparse mixture-of-experts. *arXiv preprint arXiv:2406.11353*.

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Yu Wu, and 1 others. 2024. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*.

Damai Dai, Li Dong, Shuming Ma, Bo Zheng, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Stablemoe: Stable routing strategy for mixture of experts. *Preprint*, arXiv:2204.08396.

Nishanth Dikkala, Nikhil Ghosh, Raghu Meka, Rina Panigrahy, Nikhil Vyas, and Xin Wang. 2023. On the benefits of learning to route in mixture-of-experts models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9376–9396.

Cicero Nogueira dos Santos, James Lee-Thorp, Isaac Noble, Chung-Ching Chang, and David Uthus. 2023. Memory augmented language models through mixture of word experts. *Preprint*, arXiv:2311.10768.

William Fedus, Barret Zoph, and Noam Shazeer. 2021. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *arXiv preprint arXiv:2101.03961*.

Jonathan Frankle and Michael Carbin. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2022. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*.

Yao Fu, Yinsicheng Jiang, Yeqi Huang, Ping Nie, Zhan Lu, Leyang Xue, Congjie He, Man-Kit Sit, Jilong Xue, Li Dong, and 1 others. 2024. Moe-cap: Cost-accuracy-performance benchmarking for mixture-of-experts systems. *arXiv preprint arXiv:2412.07067*.

Trevor Gale, Deepak Narayanan, Cliff Young, and Matei Zaharia. 2023. Megablocks: Efficient sparse training with mixture-of-experts. *Proceedings of Machine Learning and Systems*, 5:288–304.

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

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, and 1 others. 2023. Llama guard: Llm-based input-output safeguard for human-ai conversations. *arXiv preprint arXiv:2312.06674*.

Jean Kaddour. 2023a. The minipile challenge for data-efficient language models. *arXiv preprint arXiv:2304.08442*.

Jean Kaddour. 2023b. The minipile challenge for data-efficient language models. *Preprint*, arXiv:2304.08442.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Jakub Krajewski, Jan Ludziejewski, Kamil Adamczewski, Maciej Pióro, Michał Krutul, Szymon Antoniak, Kamil Ciebiera, Krystian Król, Tomasz Odrzygóźdź, Piotr Sankowski, and 1 others. 2024. Scaling laws for fine-grained mixture of experts. *arXiv preprint arXiv:2402.07871*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, and 1 others. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2020. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*.

Mike Lewis, Shruti Bhosale, Tim Dettmers, Naman Goyal, and Luke Zettlemoyer. 2021. Base layers: Simplifying training of large, sparse models. In *International Conference on Machine Learning*, pages 6265–6274. PMLR.

Pingzhi Li, Zhenyu Zhang, Prateek Yadav, Yi-Lin Sung, Yu Cheng, Mohit Bansal, and Tianlong Chen. 2023. Merge, then compress: Demystify efficient smoe with hints from its routing policy. *arXiv preprint arXiv:2310.01334*.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods. *Preprint*, arXiv:2109.07958.

Shuqing Luo, Pingzhi Li, Jie Peng, Yang Zhao, Yu Cao, Yu Cheng, and Tianlong Chen. 2025. Occult: Optimizing collaborative communications across experts for accelerated parallel moe training and inference. In *Forty-second International Conference on Machine Learning*.

Todor Markov, Chong Zhang, Sandhini Agarwal, Florentine Eloundou Nekoul, Theodore Lee, Steven Adler, Angela Jiang, and Lilian Weng. 2023. A holistic approach to undesired content detection in the real world. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 15009–15018.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, and 1 others. 2024. Olmoe: Open mixture-of-experts language models. *arXiv preprint arXiv:2409.02060*.

Basil Mustafa, Carlos Riquelme, Joan Puigcerver, Rodolphe Jenatton, and Neil Houlsby. 2022. Multimodal contrastive learning with limoe: the language-image mixture of experts. *Advances in Neural Information Processing Systems*, 35:9564–9576.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741.

Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. 2021. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, 34:8583–8595.

Jean Seo, Jaeyoon Kim, and Hyopil Shin. 2025. Mofe: Mixture of frozen experts architecture. *arXiv preprint arXiv:2503.06491*.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, arXiv:1701.06538.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Huaijie Wang, Lingxiao Ma, Fan Yang, Ruiping Wang, Yi Wu, and Furu Wei. 2023. Bitnet: Scaling 1-bit transformers for large language models. *arXiv preprint arXiv:2310.11453*.

Lean Wang, Huazuo Gao, Chenggang Zhao, Xu Sun, and Damai Dai. 2024. Auxiliary-loss-free load balancing strategy for mixture-of-experts. *arXiv preprint arXiv:2408.15664*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Mohan Zhang, Pingzhi Li, Jie Peng, Mufan Qiu, and Tianlong Chen. 2025. Advancing moe efficiency: A collaboration-constrained routing (c2r) strategy for better expert parallelism design. *arXiv preprint arXiv:2504.01337*.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. Lima: Less is more for alignment. *Preprint*, arXiv:2305.11206.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M Dai, Quoc V Le, James Laudon, and 1 others. 2022. Mixture-of-experts with expert choice routing. *Advances in Neural Information Processing Systems*, 35:7103–7114.

Barret Zoph. 2022. Designing effective sparse expert models. In *2022 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 1044–1044. IEEE.

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. 2022. St-moe: Designing stable and transferable sparse expert models. *arXiv preprint arXiv:2202.08906*.

A Bag of Tricks for Improving MoE Safety

A.1 Evaluation Metrics

To evaluate the safety of MoE models, we employ a set of pretrained language model-based reward and moderation metrics designed to assess harmfulness and policy alignment. Specifically, we use: ❶ **Reward Model** (Rafailov et al., 2023): A language model trained to predict human preferences regarding helpfulness, harmlessness, and honesty. It assigns higher scores to responses judged as higher possibility with harmful behavior. ❷ **Llama Guard** (Inan et al., 2023): A safety-focused classifier built on top of LLaMA, fine-tuned to detect unsafe, toxic, or policy-violating content. It provides a structured risk assessment for model outputs. ❸ **OpenAI Content Moderation API** (Markov et al., 2023): A robust, production-grade moderation tool that flags content across multiple harm categories (e.g., hate, self-harm, sexual content, and violence). The API returns category-specific scores indicating the likelihood of violation.

In all cases, higher scores reflect greater harmfulness or policy risk. These metrics enable automatic, fine-grained evaluation of the safety implications of different MoE training or routing strategies. They are especially valuable when auditing large-scale models where manual review is infeasible.

Additionally, we examine factual accuracy and robustness against hallucination using two specialized datasets. **Natural Questions (NQ)** (Kwiatkowski et al., 2019) evaluates open-domain factual accuracy. And **TruthfulQA** (Lin et al., 2022) measures the avoidance of prevalent human misconceptions.

We agree that human adjudication remains the gold standard for nuanced safety assessment. For a benchmark of this scale, however, we relied on established automated tools to ensure consistency, scalability, and reproducibility, following precedents such as MoE-RBench. Nevertheless, we acknowledge the limitations of automated detectors, including susceptibility to false positives and limited context sensitivity.

A.2 Takeways

Impact of Fine-tuning Strategies on Safety Outcomes. We evaluated the safety alignment of MoE models using various freezing fine-tuning strategies. DeepSeek MoE models consistently exhibited low safety risk metrics pre- and post-fine-tuning; thus, our analysis primarily centers on the OL-MoE series, where strategic impacts were more discernible. As illustrated in Figure 9, a strategy involving full parameter fine-tuning initially, followed by freezing the router (N→R), achieved the highest truthfulness scores and yielded the least harmful model outputs. Conversely, freezing expert parameters during the latter fine-tuning stages severely compromised model reliability and significantly increased the propensity for harmful content generation.

Consistency with Prior Findings and Validation of the N→R Sweet-Point Strategy. These safety fine-tuning outcomes align remarkably with conclusions from our prior analyses of reasoning ability (Section 3) and efficiency (Section 4). This congruence underscores the general effectiveness and versatility of the N→R "sweet point" freezing strategy in optimizing MoE models across diverse attributes, including safety.

B Additional Experiment details

B.1 Pretrain Designs

The pretrain phase plays a crucial role in establishing the foundational capabilities of MoE models. Key design decisions during this phase include the expert architecture, router design, and regularization strategies, all of which influence the model's subsequent characteristics.

B.1.1 Expert Design: Expert Granularity & Shared Experts

Expert Granularity refers to both the total number of experts in the model and the number of experts activated per input token (i.e., top-k selection). This granularity controls the sparsity level of the MoE architecture and directly affects the model's expressivity, computational cost, and specialization capacity. A larger number of experts increases the model's capacity and enables fine-grained specialization, allowing different experts to focus on distinct reasoning patterns, knowledge domains, or input types (Dai et al., 2024). However, without sufficient regularization or data diversity, this can lead to underutilization of experts. The number of activated experts per token determines how much collaboration occurs among experts. Activating more experts can enhance expressivity and redundancy but comes with more computation.

Shared Experts (Dai et al., 2024) refer to a subset of experts within a layer that are independent of

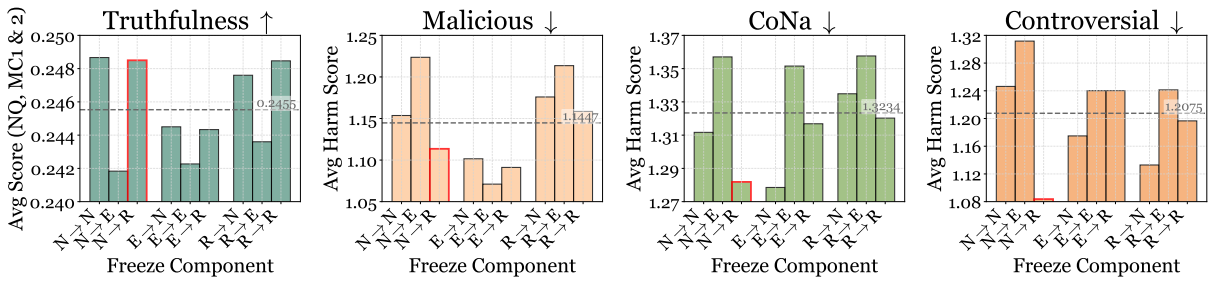

Figure 9: Effect of safety supervised fine-tuning on safety benchmarks. N→R indicates fine-tuning all parameters in the first half and freezing the router in the second half. The dashed line indicates the average performance across nine settings.

the router and always activated for all tokens. Unlike standard MoE experts selected dynamically via the router, shared experts contribute universally to every input, regardless of content or token type. This design introduces a hybrid architecture, where shared experts offer a stable transformation, while routed experts provide conditional processing. Shared experts can help stabilize training by ensuring a consistent data flow. However, too many shared experts increase the computational cost compared to fully sparse MoE architectures. Additionally, excessive reliance on shared experts can reduce the functional diversity of MoE, as these shared experts are not specialized through selective routing.

B.1.2 Router Design: Choice Strategy & Router Granularity

The router is the key control component of an MoE model, determining which experts are activated for a given input. Its design directly influences model sparsity, expert load balance, and reasoning performance. Two key dimensions are the choice strategy and the router granularity.

Choice Strategy governs how a token selects experts. Notable strategies include: ❶ **Token choice strategy** (Shazeer et al., 2017) assigns experts to tokens based on the router score. This enables token-level expert routing that adapts dynamically to input content. ❷ **Expert choice strategy** (Fedus et al., 2021) assigns tokens to experts. In this approach, each expert selects the top-k tokens it will process from a batch, forming an inverted mapping from expert to tokens. This design allows for more direct control over expert computational load by fixing the number of tokens each expert processes.

Router Granularity determines the input unit over which routing decisions are made. Different granularities include: ❶ **Token-level routing** treats each token as an independent unit for routing decisions. ❷ **Word-level routing** (dos Santos et al., 2023) computes expert assignments at the word level. Tokens belonging to the same word receive the same routing assignment, potentially reducing routing variance for semantically grouped tokens. ❸ **Sentence-level routing** extends this concept by treating all tokens within a sentence as a single unit for expert assignment.

B.1.3 Regularization Design: Load Balancing and Beyond

To ensure that experts are utilized efficiently and to prevent issues like expert starvation (where some experts are rarely chosen), regularization techniques are often employed during pre-training. These techniques aim to promote balanced expert utilization and stabilize training.

A common approach is the use of an **auxiliary load balancing loss** (Shazeer et al., 2017; Lepikhin et al., 2020; Zoph et al., 2022). This loss encourages a more uniform distribution of tokens across all available experts. For a batch of T tokens $\{x_1, \ldots, x_T\}$, the auxiliary load balancing loss L_{balance} is defined as:

$$
\begin{aligned}
L_{\text{balance}} &= \alpha \sum_{j=1}^{n} f_j P_j, \\
f_j &= \frac{1}{T} \sum_{t=1}^{T} \mathbb{I}(\text{token } x_t \text{ selected expert } j), \\
P_j &= \frac{1}{T} \sum_{s=1}^{T} (\text{softmax}(W_g x_s))_j.
\end{aligned}
\tag{2}
$$

Here, f_j represents the fraction of tokens in the batch routed to expert j, and P_j is the average router probability (i.e., the j-th component of the softmax output of the router logits $W_g x_s$ for expert j, before top-k selection) averaged over the

batch tokens x_s. The term $\mathbb{I}(\cdot)$ is the indicator function, n is the total number of experts, and α is a hyperparameter controlling the strength of this regularization. This loss is minimized when tokens are distributed uniformly and the router assigns similar probabilities to all experts on average.

Another regularization technique aimed at improving training stability is the **router z-loss** (Fedus et al., 2021). This loss penalizes large logits from the router's linear layer $W_g x$, encouraging them to be centered around zero. For a batch of T tokens, let $h(x_t) = W_g x_t$ be the vector of logits for token x_t. The router z-loss L_z is defined as:

$$L_z = \frac{1}{T} \sum_{t=1}^{T} \left(\log \sum_{j=1}^{n} e^{h(x_t)_j} \right)^2 \quad (3)$$

This penalizes the squared log-sum-exp of the logits for each token, which can prevent overly confident or extreme logit values that might destabilize training.

Beyond explicit auxiliary losses, some architectural designs inherently promote better load distribution. The **Expert Choice** routing strategy (Fedus et al., 2021) is a prime example of **auxiliary loss-free load balancing**. In this paradigm, instead of tokens choosing experts, each expert E_j selects a fixed number of tokens, c_j (its capacity), from the batch that it will process. This selection is typically based on which tokens have the highest routing scores (e.g., $(W_g x_t)_j$) for that particular expert. By design, each expert processes a predetermined number of tokens (assuming enough tokens are available in the batch), thus ensuring a balanced computational load across experts without requiring an additional loss term to enforce it.

B.2 Model Size

We evaluate several representative LLMs with varying architectures: ❶ **OLMoE**: A recent mixture-of-experts (MoE) model introduced by Allen AI in late 2024, achieving outstanding performance among 7B-scale MoE models. OLMoE is distinguished by its fully open-sourced training and fine-tuning procedures, as well as notable empirical performance. This model comprises a total of 7 billion parameters with only 1 billion parameters are activated per forward pass. It employs 8 experts per layer and uses a top-2 routing mechanism. ❷ **DeepSeek MoE**: Developed by DeepSeek, this 16B-parameter MoE model employs fine-grained expert partitioning and isolated shared experts to maximize expert specialization. It also integrates expert-level and device-level load balancing losses to mitigate routing collapse and ensure computational efficiency. It utilizes 64 experts per layer and also adopts top-2 routing.

These models illustrate the diversity in MoE architectural design, offering insights into the impact of expert structures, routing strategies, and regularization techniques.

B.3 Datasets

We conduct all pre-training experiments using the **Minipile** (Kaddour, 2023b) dataset, a high-quality corpus designed to reflect the structure and content diversity of LLM dataset Pile (Gao et al., 2020). Minipile offers a manageable scale for rapid experimentation while preserving linguistic variety and complexity, making it ideal for benchmarking architectural and training design choices. For fine-tuning, we use the **LIMA** (Zhou et al., 2023) dataset, which consists of high-quality, human-curated instruction-response pairs aimed at aligning language models with helpful and harmless behaviors. Its compact size and strong supervision signals make it well-suited for evaluating safety and alignment strategies during the fine-tuning phase.

B.4 Finetuning Hyperparameter Details

We fine-tune both OLMoE and DeepSeek MoE using the HuggingFace accelerate library and DeepSpeed Zero Stage-3. Table 1 shows the default hyperparamters of finetuning.

Hyperparameter	OLMoE / DeepSeek MoE
Precision	bf16
Attention	FlashAttention
Max Sequence Length	2048
Per-device Batch Size	1
Gradient Accu. Steps	4
Learning Rate	2e-5
Scheduler Type	Linear
Warmup Ratio	0.03
Weight Decay	0.0
Num Training Epochs	3

Table 1: Fine-tuning hyperparameter configurations for OLMoE and DeepSeek MoE models.

B.5 Evaluation Details

We evaluate the reasoning capabilities of the fine-tuned models using the EleutherAI Language Model Evaluation Harness (Gao et al., 2024). This toolkit provides a standardized interface and metric computation for a wide range of benchmark tasks.

All evaluations are conducted in a zero-shot or few-shot setting depending on the task, using consistent model interfaces.

Task	Few-shot Num	Metric
Winogrande	5	Accuracy
Hellaswag	10	Norm Accuracy
ARC-Easy	25	Accuracy
ARC-Challenge	25	Accuracy
PIQA	0	Accuracy
MMLU	5	Accuracy

Table 2: Evaluation configurations for each benchmark task used in the LM Evaluation Harness.

We follow the recommended evaluation procedure of the LM Evaluation Harness and use the respective default test sets.

B.6 Computational Resources

All experiments were conducted on a high-performance workstation running Ubuntu 20.04.6 LTS. The system is equipped with a dual-socket AMD EPYC 7763 processor, providing a total of 128 physical cores and 256 threads. For GPU acceleration, we utilized an NVIDIA RTX A6000 Ada graphics card.

C Additional Reasoning Ability Analysis

C.1 Pre-training Stage

To evaluate the impact of expert design and router configuration during the pre-training stage, we experiment with varying expert granularities (2/8 and 16/64), shared expert settings (enabled or disabled), and router designs (Token choice strategy, Expert choice strategy, Word-level routing, Sentence-level routing). Table 3 summarizes the average task accuracy across six reasoning benchmarks. Overall, our results indicate that the choice of router strategy and expert configuration significantly impacts reasoning performance, with the expert choice router consistently yielding the strongest results across various settings.

Our findings suggest that using expert-level routing and enabling shared experts consistently improves average accuracy across tasks. The highest performance (37.68%) was achieved with the 16/64 configuration, shared experts enabled, and expert-level routing.

C.2 Fine-tuning Stage

To assess the impact of component freezing strategies on reasoning ability during fine-tuning, we experiment with various freeze configurations. These include freezing the router, experts, or combinations thereof across different training phases. We also test the effect of adjusting router-specific learning rates.

Table 4 and Table 5 report the accuracy on five reasoning benchmarks for OLMoE and DeepSeek MoE, respectively. Our experiments with OLMoE show that a staged fine-tuning approach, where the router is initially unfrozen and then frozen in a later phase (None $\rightarrow$ Router), yields the highest average accuracy (65.35%).

D Additional Efficiency Analysis

D.1 Pre-training Stage

To investigate expert utilization efficiency during pre-training, we evaluate various MoE configurations using four diagnostic metrics. These metrics are designed to surface different dimensions of routing and expert dynamics: load balance, capacity, specialization, and collaboration. We evaluate these metrics across varying expert granularities as well as under ablation settings with the auxiliary loss and z-loss removed. The results are summarized in Table 6. The results indicate a clear trend where both load balance and collaboration scale with the number of available experts. Furthermore, a comparison across training settings reveals that the auxiliary loss significantly boosts these metrics, while expert specialization and capacity remain relatively stable.

D.2 Fine-tuning Stage

To better understand the efficiency behavior of MoE models under different fine-tuning configurations, we report four key metrics: load balance, capacity, specialization, and collaboration. We present results in Table 7 and Table 8 for OLMoE and DeepSeek MoE, respectively. Our analysis of fine-tuning efficiency reveals that, unlike in the pre-training stage, the expert utilization dynamics for both OLMoE and DeepSeek MoE are highly stable. While minor fluctuations exist, none of the tested fine-tuning configurations meaningfully shifted the metrics from their baseline values, indicating that the models' collaborative and specialization behaviors are well-preserved after pre-training.

E Additional Safety Analysis

To assess the safety properties of our fine-tuned MoE models, we evaluate them using multiple

Backbone	Setting	Shared	Router	ARC-C	ARC-E	MMLU	PIQA	Wino	Hella	Avg
OLMoE	2/8	Yes	Token	15.61	34.68	25.75	59.58	52.09	27.18	35.82
			Expert	18.26	37.42	25.86	**61.10**	51.54	27.09	36.88
			Word	17.92	36.53	25.91	58.92	50.20	27.24	36.12
			Sentence	18.77	36.20	26.05	57.62	50.91	26.96	36.09
		No	Token	18.00	37.24	25.44	59.96	51.22	26.80	36.44
			Expert	18.34	**38.89**	27.03	59.79	52.09	27.03	37.20
			Word	**19.80**	36.74	25.62	57.94	52.09	27.22	36.57
			Sentence	17.66	35.40	25.63	0.58	52.09	26.85	26.37
	16/64	Yes	Token	18.17	36.83	25.80	58.92	50.36	27.42	36.25
			Expert	18.43	38.80	**27.32**	60.72	**53.28**	27.52	**37.68**
			Word	18.26	36.74	25.44	59.79	50.20	27.22	36.28
			Sentence	19.37	35.02	25.88	58.43	50.99	27.28	36.16
		No	Token	17.83	35.77	25.25	58.49	52.72	27.25	36.22
			Expert	18.77	38.80	26.02	59.52	51.14	**27.54**	36.97
			Word	17.32	35.94	25.40	58.32	51.22	27.33	35.92
			Sentence	18.60	35.73	25.16	57.45	51.78	26.75	35.91

Table 3: Performance across reasoning tasks under different expert granularities (2/8, 16/64), shared expert settings (Yes/No), and router designs (Token choice strategy, Expert choice strategy, Word-level routing, Sentence-level routing). Metrics are accuracy scores on ARC Challenge (ARC-C), ARC Easy (ARC-E), MMLU, PIQA, Winogrande (Wino), and HellaSwag (Hella). Highest values in each metric column are highlighted in bold.

Model Setting	ARC-C	ARC-E	MMLU	PIQA	Wino	Avg
Baseline	46.50	78.75	51.07	80.96	68.51	65.16
Expert	46.42	78.79	50.89	80.47	68.82	65.08
Router	46.08	79.04	51.09	81.12	68.43	65.15
Expert → Router	45.90	78.70	50.68	81.01	69.22	65.10
Router → Expert	46.16	78.70	50.93	80.74	68.82	65.07
Expert → None	46.50	78.79	51.00	80.69	68.75	65.14
Router → None	46.25	79.08	51.05	81.01	68.75	65.23
None → Expert	45.73	78.58	50.80	80.41	68.11	64.73
None → Router	46.67	79.34	50.99	81.23	68.51	65.35
Router LR 1e-5	46.42	78.79	50.89	80.47	68.82	65.08
Router LR 5e-6	46.08	79.04	51.09	81.12	68.43	65.15
Router LR 2e-6	45.90	78.70	50.68	81.01	69.22	65.10

Table 4: OLMoE fine-tuning results under different freeze strategies and router learning rates. Metrics are accuracy scores on ARC Challenge (ARC-C), ARC Easy (ARC-E), MMLU, PIQA, and Winogrande (Wino). Values are presented as percentages.

Model Setting	ARC-C	ARC-E	MMLU	PIQA	Wino	Avg
Baseline	50.00	77.69	46.67	80.03	70.48	64.97
Expert	47.61	76.85	46.04	79.82	71.11	64.29
Router	50.00	77.44	46.67	80.14	70.24	64.90
None → Expert	49.32	78.03	46.45	79.82	70.88	64.90
None → Router	50.00	77.57	46.67	79.76	70.48	64.90
Expert → None	48.29	76.73	46.12	80.03	71.51	64.54
Expert → Router	48.46	77.06	46.22	79.76	71.11	64.52
Router → None	50.17	77.44	46.71	80.03	70.32	64.94
Router → Expert	49.32	77.86	46.50	79.92	70.64	64.85
Router LR 1e-5	49.57	77.99	47.17	79.92	70.64	65.06
Router LR 5e-6	47.95	77.31	47.44	80.03	71.35	64.82
Router LR 4e-6	47.01	77.65	47.26	80.03	71.19	64.63
Router LR 2e-6	47.70	77.40	47.71	79.87	70.64	64.66

Table 5: DeepSeek MoE fine-tuning results under different freeze schedules and router learning rates. Metrics are accuracy scores on ARC Challenge (ARC-C), ARC Easy (ARC-E), MMLU, PIQA, and Winogrande (Wino). Values are presented as percentages.

Model Setting	Load Balance	Capacity	Specialization	Collaboration
With Auxiliary Loss				
1/7 + 1	2.7517	0.0389	0.0532	0.0000
3/15 + 1	3.7984	0.0380	0.0579	2.9539
7/31 + 1	4.8152	0.0237	0.0521	4.0424
15/63 + 1	5.8398	0.0181	0.0511	5.0430
2/8	2.9170	0.0231	0.0603	2.0578
4/16	3.8990	0.0298	0.0512	3.1398
8/32	4.8448	0.0300	0.0556	4.1387
16/64	5.8556	0.0312	0.0487	5.1096
No Auxiliary Loss				
1/7 + 1	1.3640	0.0457	0.0489	0.0000
3/15 + 1	3.0777	0.0547	0.0589	2.3178
7/31 + 1	3.3819	0.0000	0.0015	1.2631
15/63 + 1	4.7454	0.0488	0.0520	2.9654
2/8	2.2758	0.0444	0.0546	1.4909
4/16	3.2426	0.0541	0.0508	2.6772
8/32	4.0311	0.0449	0.0484	3.2661
16/64	4.8368	0.0398	0.0419	3.0372
No z-loss				
1/7 + 1	2.8112	0.0285	0.0498	0.0000
3/15 + 1	3.8538	0.0163	0.0543	2.9225
7/31 + 1	4.6532	-0.0004	0.0008	3.5076
15/63 + 1	5.8669	0.0167	0.0463	5.0401
2/8	2.9125	0.0281	0.0536	2.0135
4/16	3.8574	0.0400	0.0611	3.1201
16/64	5.8511	0.0322	0.0590	5.0934

Table 6: Efficiency metrics under different expert granularities and training settings. $3/15+1$ denotes a configuration with 3 top experts selected from 15 routed experts, augmented by one fixed, always-activated shared expert. Metrics include load balance (token spread across experts), capacity (top-pruned delta), specialization (top vs. random output difference), and inter-expert collaboration.

Model Setting	Load Balance	Capacity	Specialization	Collaboration
Baseline	5.6884	0.0480	0.3094	4.7622
Expert	5.7009	0.0434	0.3108	4.7694
Router	5.6935	0.0462	0.3108	4.7647
Expert → Router	5.6854	0.0484	0.3083	4.7700
Router → Expert	5.6717	0.0451	0.3091	4.7572
Expert → None	5.7019	0.0384	0.3117	4.7679
Router → None	5.6897	0.0545	0.3100	4.7608
None → Expert	5.6930	0.0463	0.3054	4.7608
None → Router	5.7025	0.0362	0.3118	4.7640
Router LR 1e-5	5.7009	0.0434	0.3108	4.7694
Router LR 5e-6	5.6935	0.0462	0.3108	4.7647
Router LR 2e-6	5.6854	0.0484	0.3083	4.7700

Table 7: Efficiency metrics during fine-tuning for OLMoE under various component freezing and router LR strategies.

Model Setting	Load Balance	Capacity	Specialization	Collaboration
Baseline	5.7830	-0.0001	0.3001	4.6937
Expert	5.7700	-0.0028	0.2921	4.7080
Router	5.7830	0.0011	0.2988	4.6942
None → Expert	5.7826	-0.0005	0.2983	4.6965
None → Router	5.7827	0.0003	0.3000	4.6913
Expert → None	5.7729	-0.0012	0.2933	4.7140
Expert → Router	5.7727	0.0004	0.2956	4.7125
Router → None	5.7830	-0.0003	0.2997	4.6932
Router → Expert	5.7821	-0.0016	0.2978	4.6961
Router LR 1e-5	5.7788	-0.0006	0.2991	4.7039
Router LR 5e-6	5.7725	0.0004	0.2994	4.7073
Router LR 4e-6	5.7708	-0.0011	0.2987	4.7050
Router LR 2e-6	5.7694	-0.0025	0.2972	4.6995

Table 8: Efficiency metrics during fine-tuning for DeepSeek MoE under various component freezing and router LR settings.

safety-oriented metrics covering harmfulness, factuality, and controversial content. These evaluations help quantify how expert freezing strategies and router configurations impact model alignment and safety. We present the safety results for both OLMoE and DeepSeek MoE under a range of component freezing strategies and router-specific learning rates in Table 9 and Table 10. And we present the hallucination results in Table 11 and Table 12 Higher scores on safety metrics (Reward, LLaMA Guard, OpenAI) reflect greater harmfulness or risk, whereas lower hallucination scores are preferred. In our comprehensive safety analysis, we identify two critical findings. First, there is a substantial disparity in the baseline safety between models, with DeepSeek MoE exhibiting robust safety against harmful content generation irrespective of the fine-tuning strategy, whereas OLMoE's safety is more variable. Second, for both models, we observe a clear trade-off between certain fine-tuning approaches and factuality, as strategies involving frozen experts consistently increase the propensity for hallucination. These results underscore that the choice of fine-tuning strategy for MoE models has complex, model-dependent implications for both safety and factual reliability.

Model Setting	Malicious			CoNa			Controversial		
	Reward	LLaMA	OpenAI	Reward	LLaMA	OpenAI	Reward	LLaMA	OpenAI
Baseline	2.7631	0.5700	0.2677	2.7722	0.6629	0.5090	2.7722	0.6750	0.2917
Expert	2.7902	0.5400	0.2338	2.6914	0.7528	0.5116	2.6914	0.7000	0.3291
Router	2.7560	0.5700	0.2880	2.6736	0.6966	0.5081	2.6736	0.6500	0.2661
Expert → Router	2.7174	0.5200	0.2175	2.7549	0.7303	0.5028	2.7549	0.6900	0.3158
Router → Expert	2.8187	0.6000	0.2938	2.7527	0.7247	0.5294	2.7527	0.6500	0.3212
Expert → None	2.6698	0.5200	0.2360	2.6002	0.7079	0.4809	2.6002	0.6250	0.2994
Router → None	2.7644	0.5800	0.2778	2.5185	0.7135	0.5067	2.5185	0.6250	0.2572
None → Expert	2.8152	0.6000	0.3013	2.9170	0.7398	0.5109	2.9170	0.6750	0.3411
None → Router	2.6620	0.5700	0.2395	2.4127	0.6810	0.4926	2.4127	0.6000	0.2383
Router LR 1e-5	2.7902	0.5400	0.2338	2.6914	0.7528	0.5116	2.6914	0.7000	0.3291
Router LR 5e-6	2.7560	0.5700	0.2880	2.6736	0.6966	0.5081	2.6736	0.6500	0.2661
Router LR 2e-6	2.7174	0.5200	0.2175	2.7549	0.7303	0.5028	2.7549	0.6900	0.3158

Table 9: OLMoE safety evaluation across Malicious, CoNa, and Controversial content detection metrics using Reward Model, LLaMA Guard, and OpenAI Moderation tools. Higher scores reflect greater risk.

Model Setting	Malicious			CoNa			Controversial		
	Reward	LLaMA	OpenAI	Reward	LLaMA	OpenAI	Reward	LLaMA	OpenAI
Baseline	0.2488	0.0000	0.2863	0.1980	0.0000	0.3358	0.2153	0.0000	0.1567
Expert	0.2570	0.0000	0.2754	0.2603	0.0000	0.3441	0.2144	0.0000	0.1664
Router	0.2495	0.0000	0.2864	0.2452	0.0000	0.3351	0.2183	0.0000	0.1593
None → Expert	0.2436	0.0000	0.2938	0.2488	0.0112	0.3332	0.2012	0.0000	0.1495
None → Router	0.2431	0.0000	0.2846	0.2503	0.0000	0.3328	0.2189	0.0000	0.1502
Expert → None	0.2198	0.0300	0.2936	0.2392	0.0000	0.3411	0.1656	0.0000	0.1856
Expert → Router	0.2145	0.0300	0.2937	0.2530	0.0000	0.3384	0.1500	0.0000	0.1717
Router → None	0.2357	0.0000	0.2856	0.2378	0.0056	0.3351	0.2250	0.0000	0.1602
Router → Expert	0.2444	0.0100	0.2948	0.2550	0.0169	0.3320	0.2200	0.0250	0.1631
Router LR 1e-5	0.2216	0.0200	0.3148	0.2817	0.0000	0.3296	0.1923	0.0000	0.1423
Router LR 5e-6	0.2556	0.0000	0.2938	0.2661	0.0056	0.3345	0.2100	0.0000	0.1631
Router LR 4e-6	0.2368	0.0000	0.3014	0.2801	0.0000	0.3386	0.2021	0.0000	0.1531
Router LR 2e-6	0.2146	0.0000	0.3143	0.2286	0.0000	0.3475	0.1910	0.0000	0.1610

Table 10: DeepSeek MoE safety evaluation across Malicious, CoNa, and Controversial content detection metrics using Reward Model, LLaMA Guard, and OpenAI Moderation tools. Higher scores reflect greater risk.

Model setting	Hallucination		
	NQ	TruthfulQA MC1	TruthfulQA MC2
Baseline	0.1194	0.2521	0.3745
Expert	0.1368	0.2338	0.3562
Router	0.1197	0.2521	0.3736
Expert → Router	0.1360	0.2399	0.3571
Router → Expert	0.1186	0.2485	0.3637
Expert → None	0.1355	0.2387	0.3593
Router → None	0.1191	0.2521	0.3716
None → Expert	0.1169	0.2460	0.3626
None → Router	0.1199	0.2534	0.3722
Router LR 1e-5	0.1368	0.2338	0.3562
Router LR 5e-6	0.1197	0.2521	0.3736
Router LR 2e-6	0.1360	0.2399	0.3571

Table 11: OLMoE hallucination scores on NQ and TruthfulQA. Lower values reflect better factual alignment and fewer hallucinations.

Model setting	Hallucination		
	NQ	Truthful QA MC1	Truthful QA MC2
Bseline	0.0731	0.3207	0.4725
Expert	0.1252	0.3035	0.4529
Router	0.0734	0.3207	0.4737
None → Expert	0.0773	0.3182	0.4717
None → Router	0.0753	0.3207	0.4741
Expert → None	0.1108	0.3035	0.4509
Expert → Router	0.1158	0.3060	0.4509
Router → None	0.0759	0.3219	0.4744
Router → Expert	0.0781	0.3207	0.4713
Router LR 1e-5	0.1014	0.3195	0.4699
Router LR 5e-6	0.1133	0.3146	0.4648
Router LR 4e-6	0.1127	0.3097	0.4633
Router LR 2e-6	0.1089	0.3060	0.4638

Table 12: DeepSeek MoE hallucination scores on NQ and TruthfulQA. Lower values reflect better factual alignment and fewer hallucinations.

F Practical Trade-offs in Real-World Deployment

A comprehensive understanding of MoE models must account for the engineering trade-offs involved in deployment. While our primary focus has been evaluation, we acknowledge that latency, memory footprint, and implementation complexity are equally critical for practical adoption. Our benchmark builds on efficiency techniques such as C2R, whose systems-level implications have been carefully studied in prior work (e.g., C2R and Occult). To make this connection explicit, we summarize here the key deployment considerations highlighted in these studies and point readers to the relevant literature for in-depth analysis. This contextualization complements our evaluation by clarifying how algorithmic design choices translate into real-world engineering costs.

G Potential Risk

While MoE architectures offer significant advantages in terms of scalability and efficiency, they also present potential risks that warrant careful consideration. The capacity for MoE models to generate highly fluent and coherent text could be misused for spreading misinformation or generating other forms of harmful content if not adequately aligned with safety protocols. Future research should continue to explore robust alignment techniques and thorough evaluation methodologies specifically tailored to the unique characteristics of MoE models to address these potential risks effectively.