HMoE: Heterogeneous Mixture of Experts for Language Modeling

An Wang[1]*, Xingwu Sun[1,2]*, Ruobing Xie[1]†, Shuaipeng Li[1]
Jiaqi Zhu[1], Zhen Yang[1], Pinxue Zhao[1], Weidong Han[1]
Zhanhui Kang[1], Di Wang[1], Naoaki Okazaki[3], Cheng-zhong Xu[2]†

[1]Tencent Hunyuan, [2]University of Macau, [3]Institute of Science Tokyo

Abstract

Mixture of Experts (MoE) offers remarkable performance and computational efficiency by selectively activating subsets of model parameters. Traditionally, MoE models use homogeneous experts, each with identical capacity. However, varying complexity in input data necessitates experts with diverse capabilities, which prevents homogeneous MoE from effective expert specialization and efficient parameter utilization. In this study, we propose a novel Heterogeneous HMoE framework, where experts differ in size and thus possess diverse capacities. This heterogeneity allows for more specialized experts to handle varying token complexities more effectively. To address the imbalance in expert activation, we propose a novel training objective that encourages frequent activation of smaller experts so as to improve computational efficiency and parameter utilization. Extensive experiments demonstrate that HMoE achieves a lower loss rate with fewer activated parameters and outperforms conventional homogeneous MoE models on various pre-training evaluation benchmarks. Our codes are available at https://github.com/AnWang-AI/HMoE.

1 Introduction

Mixture of Experts (MoE) (Jacobs et al., 1991; Shazeer et al., 2017; Lepikhin et al., 2020; Fedus et al., 2022; Jiang et al., 2024; Dai et al., 2024) is a cutting-edge technique in the field of large language models (LLMs) (Brown et al., 2020; Achiam et al., 2023; Ouyang et al., 2022; Touvron et al., 2023a,b; Dubey et al., 2024) that excels in both performance and computational efficiency. At its core, MoE operates on the principle of dividing a model into multiple components, known as experts (Shazeer et al., 2017), each specializing in different tasks or aspects of the data. This specialization allows MoE to activate a subset of parameters, so as to enhance the model's robustness and flexibility. The main advantage of MoE lies in that it can scale with model parameters without incurring extra computational costs.

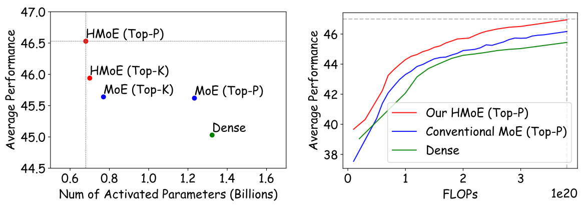


Figure 1: Comparisons of our heterogeneous MoE-3B with conventional homogeneous MoE-3B. Our proposed HMoE is superior in both performance and efficiency.

The specialization of experts (Oldfield et al., 2024; Chen et al., 2023; Krishnamurthy et al., 2023; Qiu et al., 2025) is crucial for improving computational efficiency and performance under sparse activation. However, almost all MoE models (Jiang et al., 2024; Dai et al., 2024; Wu et al., 2024; Huang et al., 2024) rely on identical experts with similar representational capacities. This design often leads to quick convergence, because experts learn similar features over time, without consideration of their uniqueness and specialization (Zhou et al., 2022; Cai et al., 2024). Such uniformity limits the model's ability to generalize effectively across tasks and undermines its performance. Moreover, the lack of functional differentiation among experts makes it challenging for MoE models to efficiently handle complex inputs in NLP (Huang et al., 2024). When all experts have equivalent representational capacities, the system fails to utilize its parameters optimally. As a result, the potential depth and diversity required for processing nuanced inputs get lost, and the effectiveness of the MoE architecture become compromised.

To address these challenges, a simple idea is to change the current homogeneous experts to het-

*These authors contributed equally.
†Corresponding authors.

erogeneous ones. **Homogeneous** indicates that all experts share identical architecture and size, while **heterogeneous** indicates that they do not. The challenges of heterogeneous MoE mainly exist in the following aspects: (a) ***How to introduce an appropriate level of heterogeneity to experts?*** This fundamental difference between homogeneous and heterogeneous MoE would significantly impact performance. (b) ***How to design and guide the desired load distributions for heterogeneous experts?*** The optimal activation of heterogeneous experts is different from that in conventional MoE. We should first conclude what kind of expert activation distribution is optimal for heterogeneous MoE, and then provide effective guidance towards such activation, balancing both parameter efficiency and model effectiveness.

In this study, we introduce a novel **HMoE** pretrained language model with varied expert sizes to create heterogeneity. We note that, without training guidance, the intuitive HMoE version does not outperform traditional MoE. Larger experts get more activation, while smaller ones are underused, reducing the model's representational capacity and hindering heterogeneous expert utilization.

We propose novel HMoE training objectives, **P-Penalty Loss**, that *encourage the activation of smaller experts*, leading to a more rational allocation of activated parameters and improved model capability. Besides, we analyze three strategies of designing different heterogeneous expert size distributions, discovering the insights of *optimal heterogeneity of experts in HMoE*.

We conduct extensive experiments to evaluate the effectiveness and efficiency of the proposed HMoE. We contribute to the success of the enhanced HMoE for the following reasons: (a) Experts of varying sizes provide diverse capacities and promote higher specialization; (b) Expert heterogeneity ensures complex inputs get the necessary resources while simpler inputs are processed economically; (c) Leveraging MoE's inherent imbalance by activating more small experts to enhance their overall capability and further reduce computing costs.

We summarize our contributions as follows:

(1) We introduce a novel HMoE model, improving both effectiveness and efficiency. To the best of our knowledge, this work should be the first to explore heterogeneous MoE as a base language model.

(2) We propose a new training objective that encourages the activation of smaller experts, leading to more efficient utilization of experts and preventing the disproportionate reliance on larger experts in HMoE.

(3) Experiments show that HMoE should perform significantly better while activating fewer parameters, thus boosting computational efficiency while enhancing downstream outcomes.

2 Methodology

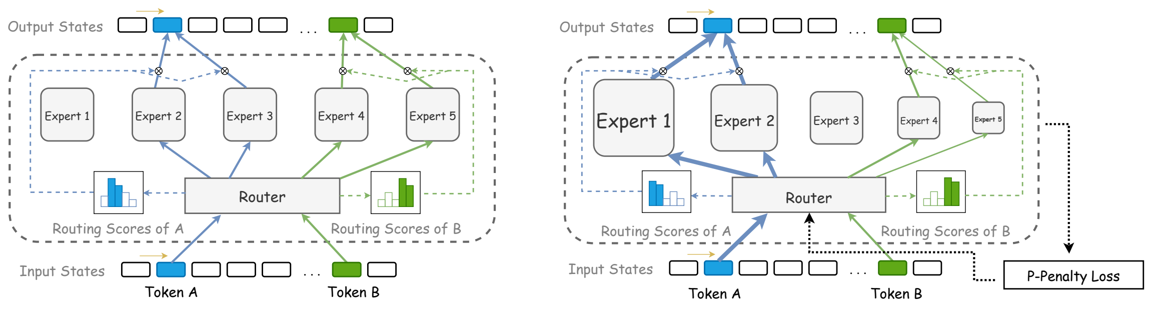
2.1 Classical Mixture of Experts

Unlike dense models, most MoE models (Lepikhin et al., 2020; Fedus et al., 2022; Huang et al., 2024; Dai et al., 2024; Jiang et al., 2024; Sun et al., 2024) replace the FFN layer of the transformer (Vaswani et al., 2017) block with a MoE layer. The MoE layer consists of a router $g_i(\cdot)$ and multiple experts $\{e_1, e_2, ..., e_N\}$. The experts are composed of a set of independent Feed-Forward Network (FFN) layers. Experts are responsible for processing input data according to their specialized knowledge. For each token, a subset of experts is activated to execute computations, and the router generates a probability distribution. The probability of this distribution indicates the likelihood of assigning the token to each expert.

Routing Strategy. The routing strategy is applied to select experts to be activated from N experts. The ***Top-K Routing*** (Shazeer et al., 2017) strategy is the most widely-used strategy, which always activates a fixed number of experts for each token. It calculates the score, which represents the probability of selecting each expert. We select the top k experts with the highest scores to activate.

Recently, ***Top-P Routing*** (Huang et al., 2024) has been proposed to dynamically activate different numbers of experts for each token. Specifically, it first sorts scores from highest to lowest. Then, given a fixed threshold p, if the highest probability is larger than the threshold, we only activate one expert. Otherwise, we progressively add additional experts until the cumulative probability exceeds the threshold p.

Issues of Homogeneous MoE. Currently, most MoE work employs a homogeneous design. Each expert in the MoE layer usually has the same structure and size. Undoubtedly, this is a simple design that avoids introducing more hyperparameters. However, it also brings the following problems: (1) Lack of Expert Specialization: Different experts within a homogeneous MoE show a tendency



(a) Conventional homogenerous MoE.

(b) Our proposed heterogeneous MoE.

Figure 2: Two distinct model structures for Mixtures of Experts are compared: (a) conventional homogeneous MoE model with all experts having identical parameter sizes; (b) our proposed heterogeneous MoE model (HMoE) characterized by substantial variations in parameter sizes of each expert, incorporating a parameter penalty loss during training to promote utilization of experts with smaller parameter volumes.

towards similarity (Zhou et al., 2022). Since homogeneous experts have the same modeling capabilities, the router may randomly distribute tokens to them during pre-training. Without differentiation mechanisms, multiple experts may focus on similar features, resulting in low specialization. Our analysis in section 3.4 shows this tendency. (2) Inefficient Parameter Allocation: Intuitively, simpler inputs can be effectively handled by smaller experts with less computational capacity, whereas more complex inputs require the enhanced capability of larger experts. However, homogeneous MoE models use experts with identical capacities, resulting in redundant computations for simple inputs and insufficient computational resources for complex ones. While Top-P Routing (Huang et al., 2024) introduces dynamic routing by assigning varying numbers of experts to different tokens, its reliance on fixed thresholds and simplistic difficulty modeling limits its ability to adapt effectively to diverse inputs. (3) Representation Collapse and Load Imbalance: Homogeneous MoE has a trend toward representation collapse (Chi et al., 2022), which occurs when the majority of input tokens are assigned to only a few experts. This phenomenon also leads to load imbalance. The interconnected nature of representation collapse and load imbalance hampers the model's performance and efficiency.

2.2 Heterogeneous Mixture of Experts

To alleviate the above issues in homogeneous MoE, we propose **Heterogeneous Mixture of Experts**. HMoE includes a router and expert network, with the key distinction that the models of experts within the same layer are different. To achieve an HMoE, we could design different structures and different sizes for experts. However, within the transformer model, experts with different structures make the training process extremely unstable. Therefore, in this work, we mainly explore HMoE with **different expert sizes**, as shown in Figure 2.

2.2.1 An Intuitive Exploration on HMoE

For each expert e_i, we follow the FFN design in LLaMa (Touvron et al., 2023a). The detailed computation is as follows:

$$e_i(\mathbf{x}) = \mathbf{W}_{o,i} \cdot (\mathrm{SiLU}(\mathbf{W}_{g,i} \cdot \mathbf{x}) \odot (\mathbf{W}_{p,i} \cdot \mathbf{x})), \tag{1}$$

$$\mathrm{SiLU}(\mathbf{z}) = \mathbf{z} \cdot \sigma(\mathbf{z}), \quad \sigma(\mathbf{z}) = \frac{1}{1+e^{-\mathbf{z}}}, \tag{2}$$

where $\mathbf{W}_{g,i} \in \mathbb{R}^{h_{\text{input}} \times h_{\text{ffn},i}}$, $\mathbf{W}_{p,i} \in \mathbb{R}^{h_{\text{input}} \times h_{\text{ffn},i}}$ and $\mathbf{W}_{o,i} \in \mathbb{R}^{h_{\text{ffn},i} \times h_{\text{input}}}$ are trainable parameters of expert e_i. h_{input} and $h_{\text{ffn},i}$ are dim of input x and hidden state in FFN. To bring in heterogeneity for exploration, we intuitively change the hidden dim $h_{\text{ffn},i}$ to control the size of each expert e_i.

2.2.2 Results of the Intuitive HMoE

We implement the aforementioned intuitive HMoE and conduct an evaluation. Contrary to our expectations, the results do not demonstrate an improvement over homogeneous MoE. Figure 3 shows the results and activation ratios of experts in HMoE.

Upon investigation, we found that the primary cause of this underperformance lies in the pronounced imbalance of load distribution among experts in the MoE model. Larger experts were activated much more frequently, while smaller ones were substantially underutilized. Importantly, this imbalance cannot be explained solely by expert size, as the relationship between size and activation rate is not strictly monotonic. The stochastic nature

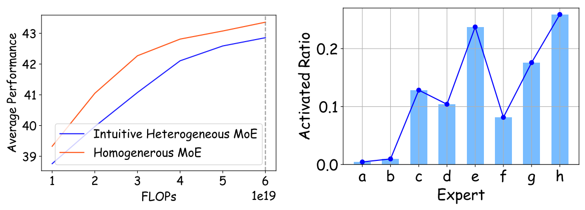


Figure 3: Experimental results of intuitive exploration on HMoE. (a) The left figure compares the results of the intuitive HMoE and conventional Homogeneous MoE. Average performance is the average score of six evaluation benchmarks as introduced in section 3.1. The Homogeneous MoE adapts load balancing loss, while the intuitive Heterogeneous MoE does not utilize any auxiliary loss. (b) The right figure shows the activated ratios of experts in the intuitive HMoE. The relative expert sizes in HMoE are $\{9, 11, 13, 15, 17, 19, 21, 23\}$, matching experts *a* to *h*.

of the router's selection process, particularly when experts are of comparable size, further contributes to this phenomenon. Ultimately, the imbalance can be traced to a Matthew effect: larger experts, having stronger representational capacity, are more likely to be favored by the router, which in turn reinforces their dominance. As a result, smaller experts receive insufficient training, thereby limiting the model's overall representational capacity.

2.3 Enhanced Heterogeneous MoE

Considering the above-mentioned issues, we propose the following strategies to enhance HMoE.

2.3.1 Activating More Small Experts

In HMoE, the presence of both large and small experts introduces a challenge where the optimization goal of the language model naturally favors the frequent activation of larger experts due to their superior performance. This tendency results in smaller experts being underutilized, while larger experts are activated more often, leading to a significant increase in activated parameters. This phenomenon diverges from the intended model objective, where we aim to align the tasks handled by large and small experts with their respective capacities. Specifically, we want larger experts to focus on more complex understanding and reasoning tasks, while smaller experts handle simpler tasks. This ensures that all specialized experts are effectively utilized and sufficiently trained according to their strengths.

Previous work (Fedus et al., 2022) adapts load balancing loss $\mathcal{L}_{\text{lb}}$ to eliminate load unbalancing among different experts in Homogeneous MoE:

$$\begin{aligned} \mathcal{L}_{\text{lb}} &= N \sum_{i=1}^{N} \mathcal{T}_i * \hat{\mathcal{P}}_i, \\ \mathcal{T}_i &= \frac{1}{T}\sum_{t=1}^{T} 1\{e_i \in E^t\}, \quad \hat{\mathcal{P}}_i = \frac{1}{T}\sum_{t=1}^{T} P_{i,t}, \end{aligned} \tag{3}$$

where $\mathcal{T}_i$ represents the partation of tokens assigned to expert e_i. $\hat{\mathcal{P}}_i$ represents the gating probability assigned to e_i. $P_{i,t}$ represents the gating probability assigned to e_i for token x_t. E^t represents the set of activated experts for the token x_t.

The objective of the load balancing loss is to achieve experts evenly activated. Nevertheless, it does not satisfy our motivation for designing HMoE. Because of the disparity in expert sizes, the load-balancing loss fails to stop the model from preferring to activate larger experts. To address the issue where larger experts are predominantly utilized, leading to the underutilization of smaller experts and a considerable rise in activated parameters, we introduce a novel training objective **parameter penalty (P-Penalty) loss** $\mathcal{L}_{\text{P-Penalty}}$ as:

$$\begin{aligned} \mathcal{L}_{\text{P-Penalty}} &= N \sum_{i=1}^{N} \mathcal{M}_i * \hat{\mathcal{P}}_i, \\ \mathcal{M}_i &= \frac{1}{T}\sum_{t=1}^{T} 1\{e_i \in E^t\} \times h_{\text{ffn},i}. \end{aligned} \tag{4}$$

$\mathcal{M}_i$ represents the average dimension of the hidden state of the expert e_i on the entire input x. It imports the influence of expert size into the loss. When the model employs more large experts, the loss rises. Hence, it will direct the model to more economically utilize smaller experts. In contrast, for necessary occasions, using larger experts can yield greater benefits than parameter penalties. At this point, larger experts will also be activated to take part in the calculation. To be noted, if all expert has the same size, our parameter penalty loss is equal to the classical load balancing loss.

Besides, with the Top-P routing strategy, we find that MoE tends to activate an increasing number of experts during training, which reduces the efficiency of MoE. Therefore, we implement the router entropy loss (Huang et al., 2024) to prevent the model from using too many parameters, maintaining its ability to selectively activate experts as:

$$\mathcal{L}_{\text{entropy}} = N \sum_{i=1}^{N} \mathrm{P}_i \times \log(\mathrm{P}_i). \tag{5}$$

In our HMoE, besides the original *language modeling loss*, the final loss for both Top-K and Top-P routing strategies further includes the *parameter penalty loss* $\mathcal{L}_{\text{P-Penalty}}$, with Top-P additionally incorporating the *router entropy loss* $\mathcal{L}_{\text{entropy}}$.

2.3.2 Designing More Optimal Heterogeneity

Intuitively, the specific sizes of each heterogeneous expert have a large impact on the final results. In this work, we mainly explore three types of heterogeneity structures for experts:

(1) ***Geometric strategy***. The geometric strategy assigns expert sizes in a geometric sequence, such as $\{1, 2, 4, 8, 16, 32, 64, 128\}$ as relative size proportions of the experts. This design emphasizes a few large-scale experts, which can lead to unbalanced resource allocation and neglect of smaller experts, potentially causing severe load imbalance and limiting its suitability for tasks requiring balanced processing.

(2) ***Arithmetic strategy***. The arithmetic strategy assigns expert sizes in an arithmetic sequence, such as $\{9, 11, 13, 15, 17, 19, 21, 23\}$. This approach can ensure balanced resource allocation and smaller size gaps between experts, giving smaller experts meaningful expressive abilities and improving training stability. This study primarily adopts this strategy for research on HMoE.

(3) ***Hybrid strategy***. The hybrid strategy that jointly combines both homogeneous and heterogeneous, such as $\{1, 1, 1, 1, 2, 2, 4, 4\}$, is also a good competitor. We designed this setup based on the assumption that the MoE model requires multiple experts with similar capabilities or functionalities. Especially in scenarios involving expert combinations, completely differentiated experts might have drawbacks. It has the flexibility to adjust the proportion of homogeneous and heterogeneous parts based on different task requirements.

As a pioneer of HMoE, we propose three strategies of different heterogeneity levels and conduct extensive evaluations in different settings for more insights. More optimal HMoE distributions and structures will be explored in the future.

3 Experiments

3.1 Experimental Settings

Pre-training Datasets. For our pre-training data, we used the RedPajama (Computer, 2023) dataset. It is an open-source dataset consisting of various sources like the common crawl, C4 (Raffel et al., 2020), GitHub, Wikipedia, books (Gao et al., 2020), arXiv, and StackExchange.

Competitors. In our main experiment, we evaluated Dense, homogeneous MoE, and our HMoE model: (1) **Dense**, which are standard Transformer decoder-only models, following the design of LLaMa (Touvron et al., 2023a), without MoE layers, implemented with 0.2B and 1B parameters. (2) Homogeneous **MoE**, where FFN layers are replaced with MoE Layers including eight homogeneous experts, implemented with 0.4B, 3B, and 16B total parameters, using both Top-K (k=2) and Top-P (p=0.6) routing strategies. (3) **HMoE**, our proposed method with Heterogeneous MoE Layers replacing FFN layers, also implemented with 0.4B, 3B, and 16B total parameters with both Top-K (k=2) and Top-P (p=0.6) strategies. To reflect the difference in performance between pure heterogeneous models and conventional homogeneous models, the expert size distribution employs an arithmetic strategy (The relative expert sizes are $\{9, 11, 13, 15, 17, 19, 21, 23\}$). The detailed setting is introduced in the Appendix A and B.

Evaluation. We evaluated these models on six different benchmarks (Gao et al., 2021) including PIQA (Bisk et al., 2020), hellaswag (Zellers et al., 2019), BoolQ (Clark et al., 2019), ARC (Clark et al., 2018), winogrande (Sakaguchi et al., 2021) and SIQA (Sap et al., 2019). These tasks examine models' language understanding, logical reasoning, knowledge utilization, and social awareness capabilities. We assessed all benchmarks under zero-shot settings. The average performance depicted in Figures 1, 3, 5, and 6 is the average score obtained across these six benchmarks. Since the activated parameters of different methods are varied, we ensure a fair comparison by basing our model evaluations on **identical computational training costs** (FLOPs) instead of the number of training tokens. Since the parameter activations of both HMoE design and Top-P routing strategy dynamically evolve during training, we compute the average activation parameters over 3 million tokens after complete training to ensure reliable comparison of dynamic activation patterns. This approach reflects the model's stable activation characteristics during inference.

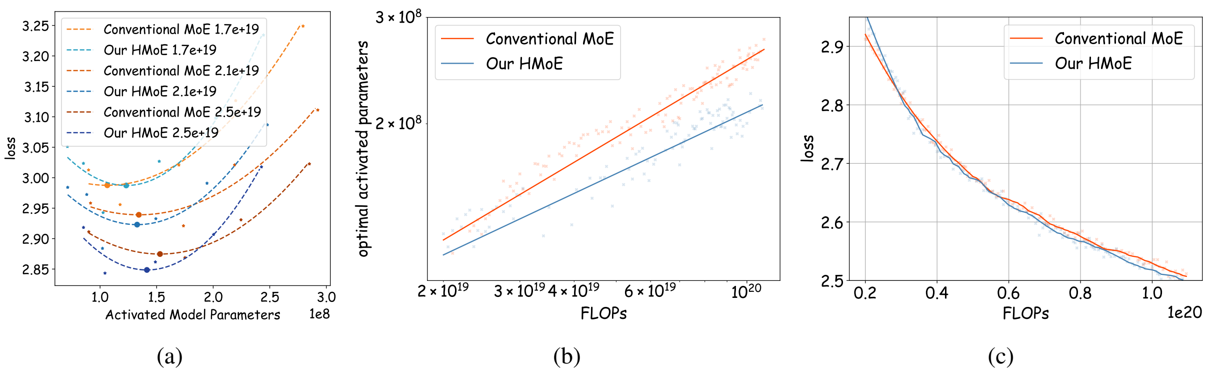


Figure 4: Analysis of isoFLOP for conventional MoE (Top-P) and our proposed HMoE (Top-P). The subfigure (a) shows examples of the activated model parameters and loss for three training FLOPs. The subfigure (b) shows the optimal activated model parameters for various training FLOPs. The subfigure (c) shows the variations in loss as FLOPs increase, given the optimal activated parameter settings.

3.2 Main Results

3.2.1 IsoFLOP Analysis

We conduct isoFLOP comparisons as shown in Figure 4. The isoFLOP analysis is a methodology used to evaluate model performance and training efficiency by fixing the training computation budget (measured in FLOPs) and comparing different model configurations. For this analysis, we adapt the TopP routing (p=0.6) strategy and train 16 conventional MoE models and 16 HMoE models of different sizes, ranging from 100M to 3B parameters, and record their activation parameters and loss values at different training FLOP levels. At each FLOP point, the activation parameter corresponding to the lowest loss is selected as the optimal activation parameter for that specific FLOP budget. This approach enables a systematic comparison of model efficiency and performance under equivalent computational constraints.

We find that if the training FLOPs are too few, the loss of HMoE is not superior to traditional MoE. However, from early stages of training (around 2.5×10^{19} FLOPs), HMoE shows a stable trend of outperforming its homogeneous MoE. Furthermore, across different training costs, the optimal activation parameter for HMoE consistently remains lower than that of homogeneous MoE. As the training cost increases, the gap in optimal activation parameters widens, highlighting the significant model efficiency advantage of HMoE. This should suggest that with larger models and more data, the benefits of heterogeneity may become even more pronounced, both in performance and efficiency.

3.2.2 Performance on Benchmarks

Table 1 presents a comparative analysis of the results of various models on benchmarks. We have:

(1) The results show MoE outperforming Dense models across all metrics, with HMoE showing particularly outstanding results. The HMoE models achieved superior performance in almost all evaluation metrics, significantly surpassing conventional MoE and Dense models.

(2) For models utilizing 7×10^{19} FLOPs, the HMoE-0.4B model, particularly with the Top-P routing strategy, stands out. It achieves an average improvement of 1.21

(3) We observe that HMoE demonstrates a more pronounced performance improvement on the ARC-Easy and HellaSwag tasks compared to conventional MoE. The rationale could be that these two tasks are comparatively easier, and P-penalty loss in HMoE is employed to guarantee sufficient training for the small experts. Meanwhile, because HMoE allocates more parameters to the larger expert, the model's performance on more challenging tasks remains uncompromised.

(4) Furthermore, the comparison between Top-K and Top-P routing within the HMoE model is also insightful. The Top-P routing strategy generally yields better results, implying that the dynamic routing strategy cooperates well with heterogeneous experts. We attribute this to the fact that both Top-P routing and heterogeneous experts are designed to adapt to the complexity of the input.

3.3 Ablation Study

We conduct an ablation study to analyze auxiliary losses and expert heterogeneity. All experiments are based on models with 400M total parameters.

Method	Activated Parameters	PIQA	hellaswag	BoolQ	ARC-Easy	winogrande	SIQA	AVG
				7×10^{19} FLOPs Training				
Dense-0.2B	176M	56.20	26.83	61.43	31.05	51.69	32.65	43.30
MoE-0.4B (Top-K)	163M	57.67	27.81	**62.13**	29.70	50.59	32.82	43.45
MoE-0.4B (Top-P)	173M	56.92	27.73	56.54	30.18	51.67	32.89	42.66
HMoE-0.4B (Top-K)	153M	56.67	**28.26**	59.80	31.93	**52.49**	**32.91**	43.68
HMoE-0.4B (Top-P)	173M	**58.98**	28.10	60.78	**34.14**	52.21	32.83	**44.51**
				2.6×10^{20} FLOPs Training				
Dense-1B	1.32B	58.92	29.57	**61.70**	35.26	51.85	32.86	45.03
MoE-3B (Top-K)	0.77B	**61.92**	32.80	60.06	33.96	52.51	32.58	45.64
MoE-3B (Top-P)	1.23B	61.42	32.16	61.47	33.51	52.27	32.91	45.62
HMoE-3B (Top-K)	0.70B	61.04	32.89	60.26	36.14	52.49	32.82	45.94
HMoE-3B (Top-P)	0.68B	61.79	**33.22**	61.69	**36.49**	**52.96**	**33.00**	**46.53**
				9×10^{20} FLOPs Training				
MoE-16B (Top-P)	3.83B	64.96	41.33	**62.56**	41.40	51.85	32.91	49.16
HMoE-16B (Top-P)	1.77B	**65.12**	**43.03**	61.40	**44.21**	**52.09**	**33.27**	**49.85**

Table 1: Results on six pre-training model evaluation benchmarks. Our HMoE consistently outperforms Homogeneous MoE. To be noted, in order to ensure a relatively fair comparison, in the experimental results of each block, although the activation parameters of different models are different, they are all trained with the same training cost (FLOPs), rather than based on the same number of training tokens.

3.3.1 Effectiveness of Auxiliary Losses

Our proposed P-Penalty loss plays a key role in HMoE's performance. To better understand the impact of auxiliary losses, we conduct an ablation study. As shown in Figure 5 (left), the P-Penalty loss helps HMoE achieve the best results among all auxiliary losses. Additionally, Figures 3 (right) and 5 (right) illustrate how auxiliary losses influence expert activation. We observe that the load balancing loss does not alleviate the tendency of larger experts being activated more frequently than smaller experts. This imbalance may limit HMoE's ability to outperform conventional MoE. In contrast, the P-Penalty loss appears to better align the model's objectives by encouraging the activation of smaller experts more frequently, thereby contributing to improved model performance and efficiency.

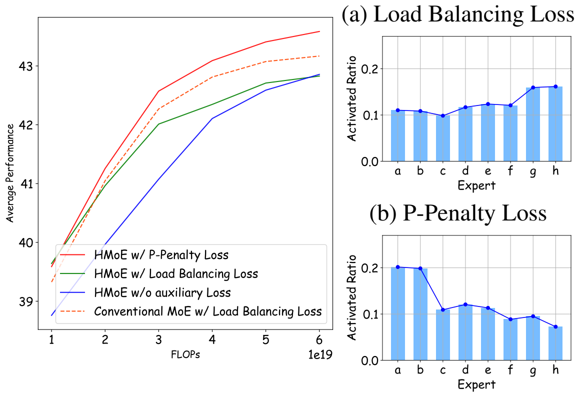


Figure 5: The left figure shows the effectiveness of auxiliary losses. The right figure shows the activated parameter ratio varying by model size across load balancing loss (above) and our P-Penalty loss (below).

3.3.2 Analyses on Expert Heterogeneity

The expert size distribution in HMoE significantly influences model performance. Figure 6 (left) compares HMoE across various distributions: geometric, arithmetic, and hybrid. Our results show that the geometric distribution performs the worst. Figure 6 (right) shows that smaller experts in the geometric progression are less frequently activated. Even with a P-Penalty loss, this may suggest their capacity is insufficient because of their too-small size. Conversely, the hybrid model outperforms the arithmetic one. This finding may indicate that a mix of experts with both similar and varied sizes offers greater potential for exploration and optimization within the HMoE model.

3.4 In-depth Analyses on HMoE Experts

To compare the expert specialization in our proposed Heterogeneous Mixture of Experts (HMoE) and traditional Homogeneous Mixture of Experts (MoE), we analyzed the behavior of experts in both setups. Figure 7 provides a similarity analysis using heatmaps, where each cell represents the Wasserstein distance between the token distributions of expert pairs on downstream tasks. In the Homogeneous MoE framework, the experts primarily cluster into two groups, suggesting limited differentiation among experts in this framework. This indicates that homogeneous setups may struggle to promote diverse expert specialization effectively.

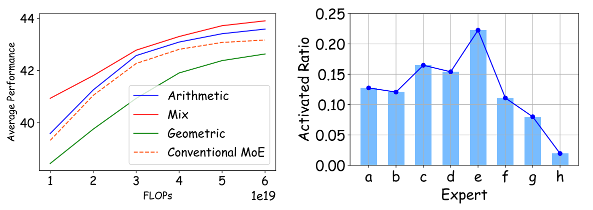


Figure 6: Analysis of expert heterogeneity through ablation. The figure on the left illustrates a performance comparison across various expert-size design strategies. The right figure displays the activation ratios of experts in HMoE using a *geometric* strategy.

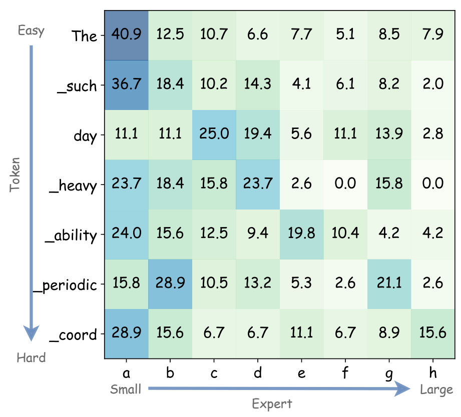
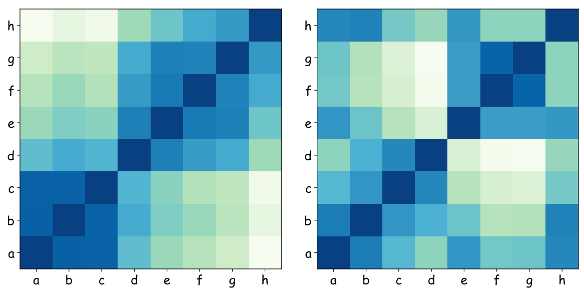


Figure 8: Visualization of the activated experts ratio to tokens with different understanding difficulty. The expert size design is the same as Figure 7.



(a) Homogeneous Experts (b) Heterogeneous Experts

Figure 7: Similarity study of homogeneous and heterogeneous experts. In the homogeneous MoE, all experts have identical sizes. In the heterogeneous MoE, the relative expert sizes are $\{9, 11, 13, 15, 17, 19, 21, 23\}$ as experts from *a* to *h*.

In contrast, the HMoE framework demonstrates a more refined expert specialization. Experts of similar sizes exhibit higher similarity, forming distinct clusters (e.g., expert pairs a/b, c/d, and f/g). This clustering may suggest that experts with comparable sizes tend to develop similar capabilities. The heterogeneous design thus encourages specialized expert behavior, emphasizing the advantages of introducing heterogeneity in fostering diversity and differentiation among experts.

Figure 8 shows the activation ratios of experts for tokens with varying difficulty levels. The activation ratio is the frequency that a token activates each expert divided by the total activations. We observe relative "hard" tokens (tokens with multiple meanings or tokens with low frequency of occurrence) activate larger experts more often, while smaller experts are consistently activated may be due to their general capabilities.

It is noteworthy that, although we present only a few examples, this phenomenon is universally observed. This should suggest that our HMoE model effectively allocates tokens to appropriate experts.

4 Related Work

The Mixture of Experts (MoE) model was first proposed by Jacobs et al. (1991), where each expert independently learns a subset of the dataset and is then integrated into a unified system. Building on this, (Shazeer et al., 2017) introduced the Sparsely-Gated Mixture-of-Experts layer (SMoE), which employs a gating network for expert selection and proposes a Top-K routing strategy, where a fixed number of experts are selected for each token. Further advancements were made by Gshard (Lepikhin et al., 2020) and SwitchTransformer (Fedus et al., 2022), which incorporated MoE into the Transformer architecture's FFN layers, utilizing top-1 and top-2 routing, respectively. Expert-choice MoE (Zhou et al., 2022) introduced Expert Choice Routing, allowing each expert to independently select a certain number of tokens, thereby achieving load balancing. AutoMoE (Jawahar et al., 2022) establishes a search space tailored for small-scale heterogeneous MoE, utilizing the top-1 routing strategy, and employs Neural Architecture Search to derive a sub-network. Their experiments focus on machine translation tasks, and their approach is not suitable for pre-trained language models. Lu et al. (2024) illustrates that not all experts are equal in the MoE model. They discard less important experts and find the model that keeps the most performance. Huang et al. (2024) introduces the Top-P routing strategy, dynamically allocating the number of experts to each token. Qiu et al. (2025) proposes a global batch with LBL for expert specialization. To be noted, our work is the first work exploring HMoE as a base language model based on Top-K and Top-P routing, and demonstrates the superiority of HMoE in both performance and efficiency.

5 Conclusion

In this work, we propose a novel HMoE model, featuring experts of varying sizes to handle different token complexities. We enhance it by proposing a new training objective and exploring expert size distribution. Our experimental results show that HMoE improves both performance and computational efficiency. We believe that our work opens new avenues for the development of large language models. Future research could explore further optimization techniques and broader applications of heterogeneous expert architectures, potentially extending the benefits observed in this study to an even wider array of NLP tasks.

6 Limitation

While our study demonstrates the substantial benefits of HMoE, several avenues for further improvement and exploration remain.

First, our experiments indicate that as training costs increase, the efficiency and performance advantages of HMoE become more evident. Although these findings suggest that HMoE could offer even greater benefits at larger scales, the precise extent of these advantages remains an open question. Second, our experiments relied on two widely adopted MoE routing strategies—Top-P and Top-K—which yielded strong performance and confirmed the broad applicability of our method. Nevertheless, the growing interest in more advanced routing approaches, such as shared experts and other dynamic mechanisms, highlights an important future direction. Our expert-size configuration is naturally complementary to such techniques, and integrating them could further enhance performance. Finally, in this study, all experts were initialized using standard schemes regardless of their size. We hypothesize that tailoring initialization to expert size may further amplify their differentiation, leading to improved diversity and effectiveness. Exploring such size-aware initialization strategies represents another promising direction for future research.

Acknowledge

Ruobing Xie is supported by the Young Elite Scientists Sponsorship Program by CAST (2023QNRC001).

References

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

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. 2024. A survey on mixture of experts. *arXiv preprint arXiv:2407.06204*.

Zitian Chen, Yikang Shen, Mingyu Ding, Zhenfang Chen, Hengshuang Zhao, Erik G Learned-Miller, and Chuang Gan. 2023. Mod-squad: Designing mixtures of experts as modular multi-task learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11828–11837.

Zewen Chi, Li Dong, Shaohan Huang, Damai Dai, Shuming Ma, Barun Patra, Saksham Singhal, Payal Bajaj, Xia Song, Xian-Ling Mao, et al. 2022. On the representation collapse of sparse mixture of experts. *Advances in Neural Information Processing Systems*, 35:34600–34613.

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. *arXiv preprint arXiv:1905.10044*.

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. *arXiv preprint arXiv:1803.05457*.

Together Computer. 2023. Redpajama: an open dataset for training large language models.

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

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39.

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, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, et al. 2021. A framework for few-shot language model evaluation. *Version v0. 0.1. Sept*, 10:8–9.

Quzhe Huang, Zhenwei An, Nan Zhuang, Mingxu Tao, Chen Zhang, Yang Jin, Kun Xu, Liwei Chen, Songfang Huang, and Yansong Feng. 2024. Harder tasks need more experts: Dynamic routing in moe models. *arXiv preprint arXiv:2403.07652*.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

Ganesh Jawahar, Subhabrata Mukherjee, Xiaodong Liu, Young Jin Kim, Muhammad Abdul-Mageed, Laks VS Lakshmanan, Ahmed Hassan Awadallah, Sebastien Bubeck, and Jianfeng Gao. 2022. Automoe: Heterogeneous mixture-of-experts with adaptive computation for efficient neural machine translation. *arXiv preprint arXiv:2210.07535*.

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, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Yechan Kim, Hwijoon Lim, and Dongsu Han. 2024. Scaling beyond the GPU memory limit for large mixture-of-experts model training. In *Forty-first International Conference on Machine Learning*.

Yamuna Krishnamurthy, Chris Watkins, and Thomas Gaertner. 2023. Improving expert specialization in mixture of experts. *arXiv preprint arXiv:2302.14703*.

Stefanos Laskaridis, Alexandros Kouris, and Nicholas D Lane. 2021. Adaptive inference through early-exit networks: Design, challenges and directions. In *Proceedings of the 5th International Workshop on Embedded and Mobile Deep Learning*, pages 1–6.

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. *Cornell University - arXiv,Cornell University - arXiv*.

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. *arXiv preprint arXiv:2402.14800*.

Xin Men, Mingyu Xu, Qingyu Zhang, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. 2024. Shortgpt: Layers in large language models are more redundant than you expect. *arXiv preprint arXiv:2403.03853*.

James Oldfield, Markos Georgopoulos, Grigorios G Chrysos, Christos Tzelepis, Yannis Panagakis, Mihalis A Nicolaou, Jiankang Deng, and Ioannis Patras. 2024. Multilinear mixture of experts: Scalable expert specialization through factorization. *arXiv preprint arXiv:2402.12550*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *NIPS-W*.

Zihan Qiu, Zeyu Huang, Bo Zheng, Kaiyue Wen, Zekun Wang, Rui Men, Ivan Titov, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. Demons in the detail: On implementing load balancing loss for training specialized mixture-of-expert models. *arXiv preprint arXiv:2501.11873*.

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(140):1–67.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social iqa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473.

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

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Hanjie Chen, et al. 2025. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*.

Xingwu Sun, Yanfeng Chen, Yiqing Huang, Ruobing Xie, Jiaqi Zhu, Kai Zhang, Shuaipeng Li, Zhen Yang, Jonny Han, Xiaobo Shu, et al. 2024. Hunyuan-large: An open-source moe model with 52 billion activated parameters by tencent. *arXiv preprint arXiv:2411.02265*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, AidanN. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Neural Information Processing Systems,Neural Information Processing Systems*.

Xun Wu, Shaohan Huang, Wenhui Wang, and Furu Wei. 2024. Multi-head mixture-of-experts. *arXiv preprint arXiv:2404.15045*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

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

A Detailed Model Setting

All methods are based on the Transformer decoder-only architecture following LLaMa (Touvron et al., 2023a). We employ the LLaMa2 (Touvron et al., 2023b) tokenizer with a vocabulary size of 32,000. We conducted a small-scale experimental exploration to determine the setting of model parameters. For the Dense-0.2B model, we configure 12 Transformer Blocks, with the hidden dimensions of the FFN layers being 3584. In the attention layer, we use 12 heads, each with a dimension of 64. For the Dense-1B model, we also configure 12 Transformer Blocks, but the hidden dimensions of the FFN layers are set to 32,768. In the attention layer, there are 16 heads, each maintaining a dimension of 64.

For both MoE (homogeneous MoE) and HMoE models, we utilize three different model sizes. (1) In the configuration with 0.4B total parameters, the total hidden dimension for all experts in each MoE layer sums to 12,288, and there are 12 Transformer Blocks. Each layer in the MoE model contains 8 experts. All other specifications align with Dense-0.4B settings. (2) In the configuration with 3B total parameters, the aggregate hidden dimension for all experts in each MoE layer is 32,768 and there are 12 Transformer Blocks. Each layer in the MoE model contains 8 experts. All other specifications match those of Dense-1B settings. (3) In the configuration with 16B total parameters, the aggregate hidden dimension for all experts in each MoE layer is 65536 and there are 40 Transformer Blocks. Each layer in the MoE model contains 16 experts. To be noted,the distribution of expert sizes in HMoE follows an **arithmetic** progression.

For Homogeneous MoE, we set the load balancing loss coefficient to 1×10^{-2}, as implemented in Huang et al. (2024). For HMoE, we set the coefficient of parameter penalty loss as 0.1. For the Top-P routing strategy, we set the coefficient of router entropy loss as 3×10^{-2}.

B Detailed Training Setting

Our models are trained utilizing NVIDIA A800 (80G memory) or H800 GPUs (80G memory). Models with fewer than 3 billion parameters are trained on a single node with 8 A800 GPUs. MoE with 16 billion parameters are trained using four nodes with a total of 32 H800 GPUs. The AdamW optimizer is used, with a first-moment decay of $\beta_1 = 0.9$ and a second-moment decay of $\beta_2 = 0.999$. A weight decay of 1×10^{-5} is applied. The learning rate is gradually increased from 0 to 1×10^{-4} over the initial 1000 steps and is maintained thereafter. The context length is set to 4096, and the global accumulated batch size is 640. All experiments use a unified random seed value of 12345. We implemented the Zero2 (Rajbhandari et al., 2020) strategy to accelerate model training and gradient checkpointing to save GPU memory. All model and training code is developed with the torch (Paszke et al., 2017) library.

C Efficient Training of Heterogeneous MoE

The efficient training of heterogeneous MoE models presents significant challenges to existing training approaches, necessitating innovative solutions to overcome these obstacles. One primary issue stems from the fact that experts do not have uniform shapes, which invalidates the traditional batched matrix multiplication method for expert computation. To address this challenge, Megablocks (Gale et al., 2023) implements efficient block sparse matrix multiplication kernels, which effectively handle the complexities introduced by variable-sized experts. Another concern is the problem of unbalanced computation and communication arising from the heterogeneous nature of experts, which can lead to inefficient resource utilization. To mitigate these issues, ES-MoE (Kim et al., 2024) introduces expert-wise offoading and dynamic expert placement strategy. This approach involves performing expert computation in a serialized manner. Expert parameters are offloaded to CPU memory and are fetched back to GPU memory as needed, based on the distribution of tokens. By doing so, ES-MoE not only reduces GPU memory overhead incurred by expert parameters but also alleviates the computation load imbalance issue, leading to better hardware resource utilization. Future research in the area may focus on developing more sophisticated load-balancing techniques and optimizing memory management strategies both for model states and activations.

D Heterogeneous Expert Parallelism

To address the load imbalance issue caused by the frequent activation of small experts in our model, we propose a new heterogeneous expert parallelism strategy. The key problem arises from the fact that small experts are activated more often than large experts, leading to an imbalance in computational load across devices. When experts of different sizes are deployed on separate devices, this imbalance results in devices hosting small experts being frequently accessed and involved in computation, while devices hosting large experts are rarely utilized, causing resource wastage.

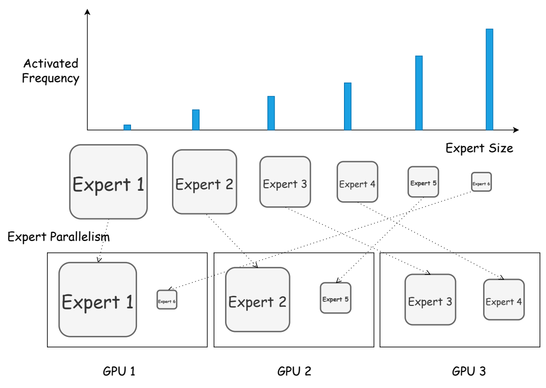


Figure 9: Example of our heterogeneous expert parallelism strategy. We integrate large experts with low activation frequency and small experts with high activation frequency into the same device to achieve load balancing.

To mitigate this issue, we place both small and large experts on the same device as shown in Figure 9, ensuring that the combined size of large and small experts on each device is approximately balanced. This strategy allows for the efficient use of GPU memory across devices by optimizing the distribution of experts. It also ensures a more balanced access frequency across devices, as the load is better distributed between devices hosting both small and large experts. Through this strategy, we can solve the efficiency problem caused by the activation imbalance in HMoE, leading to better resource utilization and overall system performance.

E Efficiency Analysis

In this section, we compare the efficiency of the heterogeneous MoE with that of the traditional homogeneous MoE model in both training and inference stages, under the condition of having the same total number of parameters and experts.

Both the heterogeneous MoE and the homogeneous MoE demonstrate similar training speeds when the total parameter count and the number of experts are kept constant as shown in Table 2. However, due to the design of the parameter penalty loss in the heterogeneous MoE, the activation parameters of the experts gradually decrease throughout the training process. Initially, the activation parameters of the heterogeneous MoE are higher than those of the homogeneous MoE, leading to a slight reduction in training speed at the beginning. As training progresses, the training speeds of both models become comparable.

Model	Training Samples per Second	Average Activated Parameters during Inference
HMoE-0.4B	13.83	153M
MoE-0.4B	13.86	163M

Table 2: Efficiency comparasion. We show the speed calculated for single A800 GPU.

During inference, the heterogeneous MoE outperforms the homogeneous MoE in terms of speed. This is because the average activation parameters in the pre-trained heterogeneous MoE are smaller, which causes the model to favor selecting smaller experts for computation. As a result, even though the number of experts selected for each inference task may be similar between the two models, the computational load per expert is reduced in the heterogeneous MoE. This leads to faster inference times compared to the homogeneous MoE, which utilizes experts of a larger size.

F Detailed Introduction of MoE

F.1 Mixture of Experts

Different from dense models, most MoE models replace the FFN layer of the transformer (Vaswani et al., 2017) block with the MoE layer. The MoE layer consists of a router $g_i(\cdot)$ and multiple experts $\{e_1, e_2, ..., e_N\}$. The experts are composed of a set of independent Feed-Forward Network (FFN) layers. Experts are responsible for processing the input data according to their specialized knowledge. For each token, a subset of experts is activated to execute computations, and the router is responsible for generating a probability distribution. The probability of this distribution indicates the likelihood of assigning the token to each expert. We obtain the output of MoE layer based on following process:

$$\begin{aligned}\text{MoE}(\mathbf{x}) &= \sum_{i}^{N} g_i(\mathbf{x}) \cdot e_i(\mathbf{x}), \\ e_i(\mathbf{x}) &= \text{FFN}_i(\mathbf{x}),\end{aligned} \tag{6}$$

where $\mathbf{x}$ is the input states of current layer.

F.2 Routing Strategy

The routing strategy is applied to select experts to be activated from N experts. The **Top-K Routing** (Huang et al., 2024) strategy is one of the most widely-used strategy, which always activates a fixed number of experts for each token. We first calculate the probability distribution $\mathbf{P}$ using a softmax function. $\mathbf{P}$ represents the initial score of selecting each expert. Then, we keep the highest k scores and normalize them. The detailed computation is as:

$$\mathbf{P} = softmax(\mathbf{W_r} \cdot \mathbf{x}) = \frac{\exp(\mathbf{W_r} \cdot \mathbf{x})}{\sum_{j=1}^{N} \exp(\mathbf{W_r} \cdot \mathbf{x})}, \tag{7}$$

$$g_i(\mathbf{x}) = \begin{cases} \frac{P_i}{\sum_{j \in \text{Top-K}(\mathbf{P})} P_j}, & i \in \text{Top-K}(\mathbf{P}) \\ 0, & i \notin \text{Top-K}(\mathbf{P}), \end{cases} \tag{8}$$

where Top-K$(\mathbf{P})$ returns the indices of the largest k elements in $\mathbf{P}$, and $\mathbf{W_r}$ is a learnable router parameter.

Recently, **Top-P Routing** (Huang et al., 2024) is proposed to dynamically activate different number of experts for each token. Specifically, we first obtain $\tilde{\mathbf{P}}$ by sorting $\mathbf{P}$ from highest to lowest. Then given a fixed threshold p, which is a hyper-parameter, if the highest probability is larger than threshold, we only use one expert. Otherwise, we progressively add additional experts until the cumulative probability exceeds the threshold p. The detailed computation is as:

$$t = \underset{k \in \{1...,N\}}{\text{argmin}} \sum_{j<=k} \tilde{\mathbf{P}}_j \geq p, \tag{9}$$

$$\text{Top-P}(\mathbf{P}) = \{\text{Index}(1), ..., \text{Index}(t)\}, \tag{10}$$

$$g_i(\mathbf{x}) = \begin{cases} \frac{P_i}{\sum_{j \in \text{Top-P}(\mathbf{P})} P_j}, & i \in \text{Top-P}(\mathbf{P}) \\ 0, & i \notin \text{Top-P}(\mathbf{P}), \end{cases} \tag{11}$$

where t represents the minimum number of experts that need to be activated. Index(j) returns the indices of element $\tilde{\mathbf{P}}_j$ in original distribution $\mathbf{P}$.

G Further Ablation on Expert Heterogeneity

Our experiments reveal a strong correlation between loss and the performance of downstream tasks: lower loss generally leads to better performance. With this insight, we investigated how to determine Expert Heterogeneity. Figure 10 illustrates the loss obtained by training HMoE using an

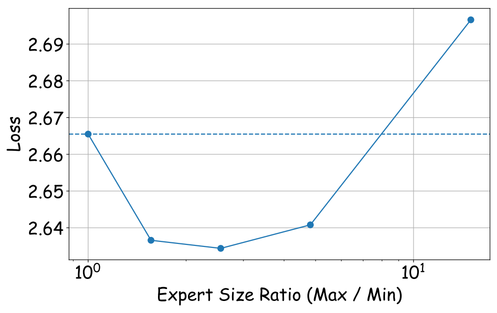


Figure 10: Various distributions of expert sizes in HMoE and their corresponding losses. All distributions follow arithmetic strategy. The x-axis represents the ratio of the size of the largest expert to the size of the smallest expert within the distribution.

arithmetic sequence strategy with varying levels of variance, all within the same computational budget. We observed that as the ratio between the largest and smallest experts increases (i.e., as the variance increases), the model's performance initially degrades but then improves. This suggests that in the heterogeneous design of HMoE, an optimal level of heterogeneity enhances performance compared to either excessive heterogeneity or complete homogeneity. This is consistent with the reason why the geometric distribution strategy has poor results. A large gap in expert ability is not conducive to model training and may lead to representation collapse. Based on these findings, we have adopted a relatively balanced heterogeneous distribution in our main experiment.

H Performance on Challenge benchmarks

We include MMLU and ARC-Challenge as additional benchmarks to further strengthen the evaluation. Under a training configuration of 2.6×10^{20}, for ARC-Challenge, the baseline SMoE-3B achieves a score of 23.41, while our HMoE-3B model achieves 25.42, showing an improvement of +2.01. For MMLU, SMoE-3B achieves 24.00, and HMoE-3B achieves 24.57, yielding an improvement of +0.57. Given that both MMLU and ARC-Challenge are relatively challenging for models operating under the current computational budget, the evaluation results are susceptible to random fluctuations. We therefore recommend referencing a comprehensive comparison incorporating the additional experimental metrics presented in this study to ensure a reliable and conclusive assessment.

I Activated Parameter Ratio Analysis

We present the activated parameter ratios of ARC tasks in HMoE layers in Table 4. Specifically, we observe that ARC-Challenge activates more parameters compared to ARC-Easy. This implies that our model can dynamically activate parameters based on the difficulty of the task. This phenomenon is consistent with that in the MoE with Top-P routing strategy (Huang et al., 2024). By activating more parameters for more difficult tasks, the model achieves better performance, while for simpler tasks, it gains higher efficiency. This approach balances efficiency and performance. To be noted, the difference in activated ratios between difficult and simple tasks is not very large, ensuring stable computational costs.

J Expert Activation Patterns

We have recorded the tokens with the highest activation percentages for different sizes of experts in the ARC tasks. As shown in Table 5, smaller experts are most frequently activated by relative simple words or words with less phonetic information. In contrast, larger experts are most frequently activated by suffix tokens. We believe that these suffix tokens may be more ambiguous and thus more difficult to understand. Medium-sized experts, on the other hand, are more frequently engaged with tokens that have clearer semantics. Importantly, this pattern emerges naturally through training rather than being intentionally designed, and the results in the table represent direct counts of activation frequency without any selective filtering.

K Activated parameters of different experts

We explore the underlying causes of the stable or declining trend in activated parameters within HMoE with Top-P routing. As depicted in Figure 11, the activation of smaller experts increases over the course of training, while larger experts experience a decline in their activation rates. This highlights the effectiveness of our proposed P-Penalty loss. The increased activation rates of smaller experts enhance their capacity to comprehend general knowledge. This shift causes the role of smaller experts to increasingly resemble that of shared experts (Dai et al., 2024). Additionally, the activation frequency of different experts remains constant throughout the training process, indicating the router's consistent token allocation.

Model	Activated Parameter Ratio	MMLU	ARC-Challenge
MoE-3B (Top-P)	1.23B	24.00	23.41
HMoE-3B (Top-P)	0.68B	24.57	25.42

Table 3: Performance of MoE and HMoE on challenge benchmarks including MMLU and ARC-Challenge under a training configuration of 2.6×10^{20} FLOPs.

Task	Activated Parameter Ratio
ARC-Challenge	21.09
ARC-Easy	20.23

Table 4: Average Activated parameter ratios (%) in HMoE layers for ARC (Clark et al., 2018) tasks.

Expert Dim	Top Tokens
2304	the, such, your, these, most, you, both, no, they, each
3328	tables, valley, sun, temper, places, day, war, water, through, clean
3840	known, least, lowest, immediately, bare, heavy, known, higher, several, independent
5376	_ly, _zen, _icker, _last, _per, _var, _orous, _next, _end, _flat
5888	_decom, _iz, _ro, _inf, _scra, _coord, _er, problem, _och, _foss

Table 5: Top activated tokens for each expert.

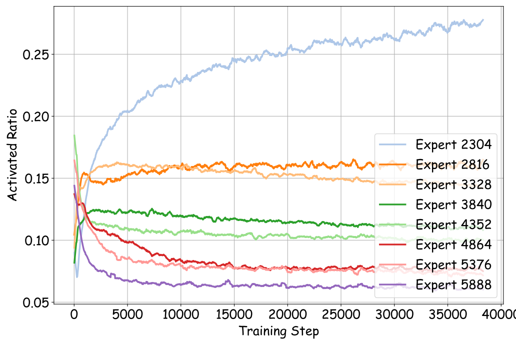


Figure 11: Activated parameters of experts in HMoE (Top-P). The values in the legend indicate the hidden dimensions of the experts, which represent their sizes.

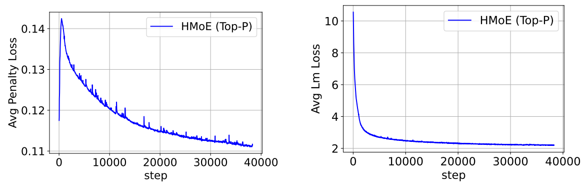


Figure 12: Training losses: (a) P-Penalty Loss, (b) Language modeling Loss.

L P-Penalty Loss during Training

This work proposes P-Penalty Loss to adjust the activation changes of experts of different sizes. To demonstrate the effectiveness of P-Penalty Loss, we show its changes during training in Figure 12. As training progresses, the language modeling loss continues to decline, while the P-Penalty Loss rises rapidly in the first 1,000 steps (approximately 3B tokens) before gradually decreasing. This is because larger experts, due to their stronger expressive capabilities, yield a greater reduction in LM loss from activation compared to the penalty imposed by P-Penalty Loss.

M Compatibility with Low-cost Inference Mechanisms

A concern regarding heterogeneous MoE is whether allocating parameters to smaller experts inherently weakens the representation ability of the model. In fact, the proposed HMoE does not reduce the total number of expert parameters but redistributes them across experts of different sizes. Thus, in theory, the total representation capacity is preserved. The heterogeneous distribution mainly influences the dynamics of expert activation rather than the overall parametric budget.

It is also worth noting that HMoE is compatible with other low-cost inference mechanisms, such as dynamic early-exit (Laskaridis et al., 2021; Sui et al., 2025) and pruning strategies (Men et al., 2024). These approaches aim to reduce the computational cost at inference time, without modifying the underlying expert structures during training. In contrast, HMoE focuses on structural optimization during model design. Therefore, the two directions are complementary rather than mutually exclusive.