MIXTURE OF NEURON EXPERTS

Anonymous authors
Paper under double-blind review

ABSTRACT

In this work, We first explore whether the parameters activated by the MoE layer remain highly sparse at inference. We perform a sparsification study on several representative MoE models. For each expert, we rank parameters by the magnitude of their activations from the gate projection and progressively prune the activated subset. Pruning up to 60% of parameters within that subset causes only negligible task-performance degradation; substantial drops occur only after more than 90% are removed. We further decompose experts into neuron granular MoE and visualize their activation values, finding that most neuron activations are near zero. This observation motivates us to select only high-activation neuron experts during pretraining. Based on this insight, we propose *Mixture of Neuron Experts* (MoNE). MoNE achieve neuron granular expert selection by only applying a simple top-k selection within each expert, incurs negligible latency, and requires no additional routing parameters or inter-expert communication. Extensive experiments demonstrate that MoNE matches traditional MoE performance while activating only 50% of the MoE-layer parameters, and it consistently outperforms traditional MoE when compared at equal numbers of activated parameters. These results suggest that MoNE is a practical approach to improving parameter utilization and inference efficiency in MoE-like models.

1 INTRODUCTION

Large language models (LLMs) (Dai et al., 2024; Bai et al., 2023; Agarwal et al., 2025; Team et al., 2025) based on Mixture-of-Experts approaches have attracted growing interest in both academic research and industry. The fundamental concept of Mixture-of-Experts (MoE) in large language models entails partitioning a large feed-forward network (FFN) into several smaller subnetworks referred to as experts, where only a subset of expert parameters are activated depending on the input. Unlike dense models that activate all parameters uniformly, MoE models achieve greater computational efficiency through sparse activation patterns.

One key motivation for MoE is the long-observed activation sparsity (Frankle & Carbin, 2018; Fedus et al., 2022a; Frantar & Alistarh, 2023; Frankle et al., 2019) in dense networks: for an given input, only a small fraction of parameters are effectively used. Mixture-of-Experts architectures exploit this property via conditional computation, activating a sparse subset of specialists so as to substantially increase model capacity while maintaining computational efficiency. This further motivates us to propose the following question:

Are the parameters activated by the MoE layer still highly sparse at inference?

To answer the question, we performed a sparsification study on a set of representative MoE models. For each expert, we ranked the parameters according to the magnitude of their activations values, which calculated by the gate projection. Then we progressively pruned the weights from the activated subset according to their rank. The results are presented in Figure 1. Notably, across three evaluated models, removing up to 60% of the parameters in this subset led to only negligible declines in task performance, with significant degradation occurring only after more than 90% were pruned. These results suggest that the parameter subset selected by the MoE gating mechanism still highly sparsity at inference.

To further explore the sparsity of MoE, we decompose expert into neuron granular MoE. Then we visualize the activation value for the neuron experts. As shown in Figure 2, most of the activation

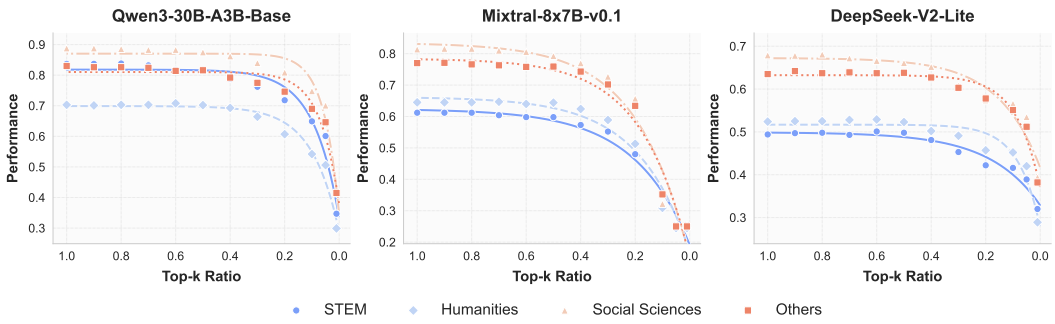

Figure 1: The performance of mainstream MoE models when only use the neuron experts with higher activation weight without extra training. Top-K Ratio refers to the ratio of selected neuron experts.

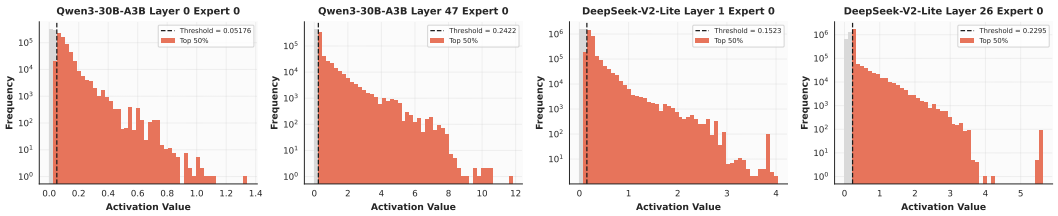

Figure 2: The activation value for the neuron experts, and the top 50% of these values were highlighted.

values are small, which further demonstrate the sparsity of the MoE layer. This results motivate us to only use the neuron experts with high activation weights for pretraining. Also, recently studies have shown the importance of expert granularity (Krajewski et al., 2024; Lepikhin et al., 2020; Du et al., 2022): Deepseek V3 Liu et al. (2024) applies 256 experts, Kimi K2 Team et al. (2025) applies 384 experts, and Qwen3-NextTeam (2025) applies 512 experts. However, overly fine-grained expert partitioning requires substantially larger routing networks and incurs significant cross-device communication latency (Lepikhin et al., 2020; Fedus et al., 2022b). Therefore, we propose Mixture of Neuron Experts (MoNE). By applying a simple top-k selection within each expert, we achieve granular selection for MoE without introducing additional router parameters or inter-expert communication. We evaluate MoNE under multiple settings and find that it consistently outperforms traditional MoE.

Our contributions are summarized as follows:

- We emprically show that traditionally trained Mixture-of-Experts models exhibit high activation sparsity at inference: a small subset of parameters with large activation values retains most of the model's capability, and most of the activation values are small in the MoE layer.
- We introduce *Mixture of Neuron Experts* (MoNE), which decompose the expert into neuron granular MoE, and achieve neuron granular expert selection via a simple top-k within-expert operation that incurs negligible latency overhead and requires no additional routing parameters.
- Extensive experiments demonstrate that MoNE matches the performance of traditional MoE while using only 50% of the parameters in MoE layer. With the same total number of activated parameters, MoNE consistently outperforms traditional MoE.

2 Related Work

2.1 Large Language Models

Large language models (LLM) (Touvron et al., 2023a; Bai et al., 2023; Brown et al., 2020; Achiam et al., 2023; Liu et al., 2024; Devlin et al., 2019; Raffel et al., 2020) have shown remarkable abil-

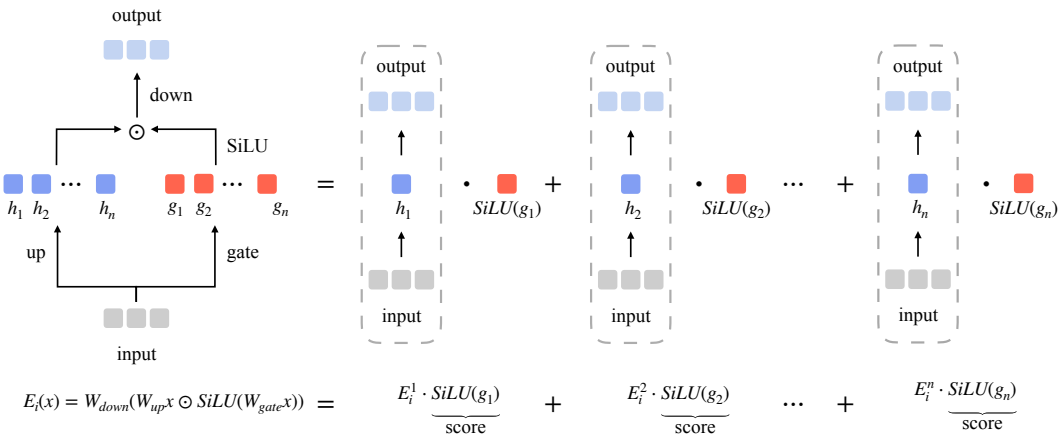

Figure 3: Expert in traditional MoE can be decomposed as the weighted sum of neuron granular FFN, which can be realized as a neuron granular MoE.

ities across different tasks, representing important progress toward artificial general intelligence. This success is largely driven by the growth of training data and the expansion of model parameter counts(Wei et al., 2022; Kaplan et al., 2020). And many recent works successfully scaling LLM to billions of parameters (Dai et al., 2024; Liu et al., 2024; Team et al., 2025; Agarwal et al., 2025; Zhang et al., 2022; Scao et al., 2022). However, As model scale increases, the demand for computational resources rises sharply. Consequently, improving the efficiency of both training and inference has become a central research focus to enable further scaling of large language models.

2.2 Mixture of Experts

The Mixture of Experts (MoE) (Cai et al., 2025; Masoudnia & Ebrahimpour, 2014; Jiang et al., 2024) architecture was introduced to enhance the capacity of deep neural networks while maintaining computational efficiency. Shazeer et al. (2017) proposed integrating an MoE layer between LSTM layers, demonstrating strong performance in language modeling and machine translation tasks. This approach was later adapted into the transformer framework by replacing the standard feed-forward layers with MoE modules. The Switch Transformer (Fedus et al., 2022b) streamlines the expert selection process by assigning each token to only the top-ranked expert, enabling more efficient model scaling. Gshard (Lepikhin et al., 2020) refined the routing mechanism by employing a Top-2 expert strategy, leading to substantial improvements in multilingual translation across 100 languages. More recently, DeepseekMoE (Dai et al., 2024) and have introduced fine-grained partitioning of experts within the MoE structure. Grove-MoE (Wu et al., 2025) incorporating experts of varying sizes. PEER (He, 2024) scales the number of experts up to one million. Kimi-K2 (Team et al., 2025) scales the model to 1,000B parameters and employs 384 experts, while Qwen3-Next (Team, 2025) further increases the expert count to 512. Improving expert granularity (Tian et al., 2025; Krajewski et al., 2024) and increasing the utilization of activated parameters (Li et al., 2025) are central goals in contemporary MoE research. However, overly fine-grained expert partitioning requires substantially larger routing networks and incurs significant cross-device communication overhead and latency (Lepikhin et al., 2020; Fedus et al., 2022b). PKM (Lample et al., 2019) achieves fast retrieval in a key-value memory layer using a structured memory architecture and further scaled by (Huang et al., 2024). In this work, we analyze the sparsity of activated parameters in traditional MoE architectures and construct a neuron granularity MoE model. This neuron granularity conversion substantially improves the utilization of activated parameters while avoiding the need for a large router and communication latency associated with overly fine expert partitioning in traditional MoE.

3 METHOD

3.1 PRELIMINARY ON MOE

The Mixture-of-Experts (MoE) layer extends a standard Transformer by replacing a dense feed-forward network (FFN) with a collection of expert FFNs and a routing mechanism. For each input token, the router conditionally dispatches only a sparse subset of experts, so that computation is performed by a small number of specialists rather than the entire FFN. This sparse execution increases the model's effective capacity while keeping per-token computation and latency approximately constant, enabling parameter-efficient scaling to much larger models. (Lepikhin et al., 2020; Dai et al., 2024; Fedus et al., 2022b).

Formally, let $\mathbf{x} \in \mathbb{R}^{d_{\text{model}}}$ denote an input hidden state. An MoE layer consists of N_E experts $\{E_i\}_{i=1}^{\mathrm{N}_E}$ together with a router that maps $\mathbf{x}$ to routing logits. The experts typically use the *Gated Linear Unit* (Dauphin et al., 2017) structure, which can be formulated as:

$$\mathbf{E}_i(\mathbf{x}) = \mathbf{W}^i_{\text{down}}(\texttt{SiLU}(\mathbf{W}^i_{\text{gate}}\mathbf{x}) \odot \mathbf{W}^i_{\text{up}}\mathbf{x}) \tag{1}$$

where $\mathbf{W}^i_{\text{gate}} \in \mathbb{R}^{d_{\text{model}} \times d_{\text{expert}}}$ is the gate projection, $\mathbf{W}^i_{\text{up}} \in \mathbb{R}^{d_{\text{model}} \times d_{\text{expert}}}$ is the up projection, and $\mathbf{W}^i_{\text{down}} \in \mathbb{R}^{d_{\text{expert}} \times d_{\text{model}}}$ is the down projection. A common routing pipeline is to first calcuate the scores of the router:

$$\mathbf{P}(\mathbf{x}) \;=\; \texttt{Act}(\texttt{topK}(\text{Router}(\mathbf{x}))), \tag{2}$$

where $\texttt{topK}(\cdot)$ masks out all but the top-K routing logits and $\texttt{Act}(\cdot)$ is the activation function. Then the MoE output is calculated as follows:

$$\text{MoE}(\mathbf{x}) \;=\; \sum_{i=1}^{\mathrm{N}_E} \mathbf{P}(\mathbf{x})_i \, \mathbf{E}_i(\mathbf{x}). \tag{3}$$

To encourage balanced utilization across experts, we adopt the commonly used auxiliary load-balance loss:

$$\mathcal{L}_{\text{aux}} \;= \alpha_{\text{aux}} \cdot \mathrm{N}_E \cdot \sum_{i=1}^{\mathrm{N}_E} \mathbf{f}_i \cdot \mathbf{P}_i, \quad \text{where} \tag{4}$$

$$\mathbf{f}_i = \frac{1}{\mathrm{T}} \sum_{\mathbf{x} \in \mathcal{B}} \mathbb{1}\{i \in \texttt{argtopK}(\text{Router}(\mathbf{x}))\}, \quad \mathbf{P}_i = \frac{1}{\mathrm{T}} \sum_{\mathbf{x} \in \mathcal{B}} \frac{\texttt{Act}(\text{Router}(\mathbf{x}))[i]}{\sum_{t=1}^{\mathrm{N}_E} \texttt{Act}(\text{Router}(\mathbf{x}))[t]} \tag{5}$$

Here, $\texttt{argtopK}$ get the index of the top-K routing logits, $\mathbf{f}_i$ is the fraction of tokens in the batch $\mathcal{B}$ that are assigned to expert i, and $\mathbf{P}_i$ is the average gating weight that the router assigns to expert i (both estimated over a batch of size T). Minimizing $\mathcal{L}_{\text{aux}}$ therefore penalizes experts that are either under-selected or consistently receive low gating weight, encouraging the router to distribute token assignments and gating weights more evenly across experts. The coefficient α_{aux} controls the regularization strength, and the multiplicative factor N_E normalizes the objective with respect to the number of experts.

3.2 MIXTURE OF NEURON EXPERTS

To explore the sparsity within the activated experts, we decompose the each experts into neuron granular mixture of experts. The output of an expert is formulated as:

$$\mathbf{E}_i(\mathbf{x}) = \mathbf{W}^i_{\text{down}}(\texttt{SiLU}(\mathbf{W}^i_{\text{gate}}\mathbf{x}) \odot \mathbf{W}^i_{\text{up}}\mathbf{x}) \tag{6}$$

For clarity and compactness, we denote the outputs of the gate projection and the up projection by $\mathbf{G}$ and $\mathbf{H}$, respectively. Concretely,

$$\mathbf{G} \;=\; \texttt{SiLU}(\mathbf{W}^i_{\text{gate}}\mathbf{x}) \in \mathbb{R}^{d_{\text{expert}}}, \quad \mathbf{H} \;=\; \mathbf{W}^i_{\text{up}}\mathbf{x} \in \mathbb{R}^{d_{\text{expert}}}. \tag{7}$$

Then Equation (6) can be reformulated as:

$$\mathbf{E}_i(\mathbf{x}) \;=\; \mathbf{W}^i_{\text{down}}(\mathbf{G} \odot \mathbf{H}). \tag{8}$$

Algorithm 1: Mixture of Neuron Experts (MoNE)

Input: Layer input $\mathbf{x} \in \mathbb{R}^{d_{\text{model}}}$, number of experts n, number of selected experts K_E, number of selected neurons for each expert K_N, activation function of router `Act`,
Output: Layer output $\mathbf{h} \in \mathbb{R}^{d_{\text{model}}}$
▷ Initialize $\mathbf{h} = \mathbf{0}$
$\mathbf{p} = \mathrm{Router}(\mathbf{x}) \in \mathbb{R}^n$ // Calculate the scores for each expert
$\mathbf{I}_E = \texttt{argtopK}(\mathbf{p})$ // Select top-K_E experts
$\hat{\mathbf{p}} = \texttt{Act}(\mathbf{p}[\mathbf{I}_E])$ // Calculate the activated scores
for each selected expert $i \in \mathbf{I}_E$ **do**
  $\mathbf{G}_i = \texttt{SiLU}(\mathbf{W}^i_{\text{gate}}\mathbf{x})$ // Calculate the output of down projection
  $\mathbf{I}_N = \texttt{argtopK}(\texttt{Abs}(\mathbf{G}_i))$ // Select top-K_N neurons
  $\tilde{\mathbf{W}}^i_{\text{up}} = \mathbf{W}^i_{\text{up}}[\mathbf{I}_N, :]$, $\tilde{\mathbf{W}}^i_{\text{down}} = \mathbf{W}^i_{\text{down}}[:, \mathbf{I}_N]$ // Select the weights used for calculation
  $\tilde{\mathbf{E}}_i(\mathbf{x}) = \tilde{\mathbf{W}}^i_{\text{down}}(\mathbf{G}_i[\mathbf{I}_N] \odot \tilde{\mathbf{W}}^i_{\text{up}}\mathbf{x})$ // Calculate the output of expert i
  $\mathbf{h} = \mathbf{h} + \hat{\mathbf{p}}[i] \cdot \tilde{\mathbf{W}}^i_{\text{up}}\mathbf{x}$ // Sum the layer output
end for
return $\mathbf{h}$

Let $\mathbf{W}^i_{\text{down}}[:, k] \in \mathbb{R}^{d_{\text{model}}}$ denote the k-th column of $\mathbf{W}^i_{\text{down}}$ and $\mathbf{W}^i_{\text{up}}[k, :] \in \mathbb{R}^{1 \times d_{\text{model}}}$ the k-th row of $\mathbf{W}_{\text{up}}$, then we expand the product as follows:

$$\mathbf{E}_i(\mathbf{x}) = \sum_{k=1}^{d_{\text{expert}}} \mathbf{W}^i_{\text{down}}[:, k] \left(\mathbf{G}[k] \cdot \mathbf{H}[k]\right) \tag{9}$$

$$= \sum_{k=1}^{d_{\text{expert}}} \mathbf{G}[k] \cdot \left(\mathbf{W}^i_{\text{down}}[:, k] \left(\mathbf{W}^i_{\text{up}}[k, :]\mathbf{x}\right)\right) \tag{10}$$

$$= \sum_{k=1}^{d_{\text{expert}}} \mathbf{G}[k] \cdot \Big(\underbrace{\left(\mathbf{W}^i_{\text{down}}[:, k]\mathbf{W}^i_{\text{up}}[k, :]\right)}_{\mathbf{A}_k} \mathbf{x}\Big). \tag{11}$$

Consequently, we can get the decomposition as follows:

$$\mathbf{E}_i(\mathbf{x}) \;=\; \sum_{k=1}^{d_{\text{expert}}} \mathbf{G}[k] \cdot \mathbf{A}_k\mathbf{x}\,, \quad \text{where} \quad \mathbf{A}_k \;=\; \mathbf{W}^i_{\text{down}}[:, k]\mathbf{W}^i_{\text{up}}[k, :] \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}} \tag{12}$$

Eq. equation 12 shows that each expert can be decomposed into a set of neuron granular experts $\mathbf{A}_k$ that weighted by the neuron level activations $\mathbf{G}$. (**In the subsequent articles, we refer to traditional experts as experts and to neuron granular experts as neuron experts.**) Motivated by this perspective, we explore the distribution of $\mathbf{G}$ in the mainstream MoE models at inference. As shown in Figure 2, the majority of neuron experts receive negligible gate weights: most values of $\mathbf{G}$ are very small, indicating that a large fraction of neurons inside each expert are inactive during inference. To further quantify the impact of these low-activation neurons, we ablate neuron experts whose gate values fall below a threshold and measure the resulting performance. As shown in Figure 1 we find that retaining approximately the top 30% of neuron experts by gate magnitude is sufficient to preserve the bulk of the original performance, which implies that conventionally trained MoE architectures induce high sparsity in the set of activated parameters at inference.

To address this issue, we propose the *Mixture of Neuron Experts* (MoNE). Algorithm 1 formulates the pipeline of MoNE. Concretely, the router first selects a set of experts as usual; for each selected expert i, we first calculate the the neuron gating weights $\mathbf{G}$, then we use the neuron experts associated with high absolute gating values to calcuate the ouput of each experts. MoNE converts the traditional MoE into a neuron granular MoE via a simple single, per-expert sort-and-select operation. In contrast, an explicit neuron level routing design in traditional MoE would require a substantially larger router and incur heavy cross-expert communication overhead (Lepikhin et al., 2020). MoNE's selection incurs no additional routing parameters and only accesses the parameters of the expert itself ; because the selected neuron experts communicate within their host expert, the

Table 1: Comparison between traditonal MoE and MoNE with the same number of activated experts. MoNE shows comparable results while only use half of the activated parameters in the MoE Layer.

Model	ARC-C	BOOIQ	HELLA	LAMBDA	MNLI	PIQA	RACE	SIQA	WINO	WNLI	AVG.
Traditional MoE 4E/64E	$30.55_{\pm 1.35}$	$56.94_{\pm 0.87}$	$47.78_{\pm 0.50}$	$32.70_{\pm 0.65}$	$34.39_{\pm 0.48}$	$69.53_{\pm 0.78}$	$30.33_{\pm 0.65}$	$39.87_{\pm 1.11}$	$52.80_{\pm 1.19}$	$40.85_{\pm 1.69}$	43.57
MoNE w/ Random Selection 4E/64E	$27.39_{\pm 1.30}$	$56.79_{\pm 0.87}$	$37.94_{\pm 0.48}$	$27.56_{\pm 0.62}$	$35.49_{\pm 0.48}$	$65.13_{\pm 0.81}$	$30.24_{\pm 0.65}$	$38.84_{\pm 1.10}$	$49.88_{\pm 1.19}$	$43.66_{\pm 1.70}$	42.69
MoNE w/ TopK Selection 4E/64E	$31.31_{\pm 1.35}$	$53.64_{\pm 0.88}$	$48.08_{\pm 0.50}$	$34.76_{\pm 0.66}$	$33.96_{\pm 0.48}$	$70.02_{\pm 0.78}$	$31.48_{\pm 0.66}$	$39.25_{\pm 1.10}$	$51.78_{\pm 1.19}$	$47.89_{\pm 1.71}$	**44.21**

Table 2: Comparison between traditonal MoE and MoNE with the same number of activated parameters. For 920M parameter and 2.8M LLMs, MoNE exhibits better downstream performance than MoE models.

Model	ARC-C	BOOIQ	HELLA	LAMBDA	MNLI	PIQA	RACE	SIQA	WINO	WNLI	AVG.
925M Activated 925M											
Dense	$33.28_{\pm 1.38}$	$58.62_{\pm 0.86}$	$52.07_{\pm 0.50}$	$37.05_{\pm 0.67}$	$33.49_{\pm 0.48}$	$71.27_{\pm 0.77}$	$30.72_{\pm 0.66}$	$40.99_{\pm 1.11}$	$54.22_{\pm 1.19}$	$52.11_{\pm 1.71}$	46.38
925M Activated 290M											
Traditional MoE 4E/64E	$30.55_{\pm 1.35}$	$56.94_{\pm 0.87}$	$47.78_{\pm 0.50}$	$32.70_{\pm 0.65}$	$34.39_{\pm 0.48}$	$69.53_{\pm 0.78}$	$30.33_{\pm 0.65}$	$39.87_{\pm 1.11}$	$52.80_{\pm 1.19}$	$40.85_{\pm 1.69}$	43.57
MoNE w/o NG-LBL 8E/64E	$30.97_{\pm 1.35}$	$55.75_{\pm 0.87}$	$48.01_{\pm 0.50}$	$33.34_{\pm 0.66}$	$34.44_{\pm 0.48}$	$70.67_{\pm 0.78}$	$29.86_{\pm 0.65}$	$38.89_{\pm 1.10}$	$53.83_{\pm 1.19}$	$49.30_{\pm 1.72}$	**44.51**
MoNE w/ NG-LBL 8E/64E	$30.38_{\pm 1.34}$	$59.45_{\pm 0.86}$	$49.51_{\pm 0.50}$	$33.96_{\pm 0.66}$	$34.79_{\pm 0.48}$	$70.89_{\pm 0.77}$	$30.24_{\pm 0.65}$	$39.76_{\pm 1.11}$	$53.67_{\pm 1.19}$	$52.11_{\pm 1.71}$	**45.48**
925M Activated 310M											
Traditional MoE 6E/64E	$33.02_{\pm 1.37}$	$54.50_{\pm 0.87}$	$49.20_{\pm 0.50}$	$34.48_{\pm 0.66}$	$35.04_{\pm 0.48}$	$71.33_{\pm 0.77}$	$30.91_{\pm 0.66}$	$40.58_{\pm 1.11}$	$53.43_{\pm 1.19}$	$36.62_{\pm 1.65}$	43.91
MoNE w/o NG-LBL 12E/64E	$30.97_{\pm 1.35}$	$55.99_{\pm 0.87}$	$50.07_{\pm 0.50}$	$35.73_{\pm 0.67}$	$32.35_{\pm 0.47}$	$69.97_{\pm 0.78}$	$30.81_{\pm 0.66}$	$40.43_{\pm 1.11}$	$51.78_{\pm 1.19}$	$52.11_{\pm 1.71}$	**45.02**
MoNE w/ NG-LBL 12E/64E	$32.17_{\pm 1.36}$	$62.11_{\pm 0.85}$	$48.65_{\pm 0.50}$	$34.58_{\pm 0.66}$	$33.89_{\pm 0.48}$	$71.44_{\pm 0.77}$	$31.00_{\pm 0.66}$	$39.20_{\pm 1.10}$	$52.88_{\pm 1.19}$	$50.70_{\pm 1.72}$	**45.75**
925M Activated 330M											
Traditional MoE 8E/64E	$32.68_{\pm 1.37}$	$56.61_{\pm 0.87}$	$49.70_{\pm 0.50}$	$35.51_{\pm 0.67}$	$33.36_{\pm 0.48}$	$71.93_{\pm 0.77}$	$30.33_{\pm 0.65}$	$40.74_{\pm 1.11}$	$51.22_{\pm 1.19}$	$39.44_{\pm 1.68}$	44.30
MoNE w/o NG-LBL 16E/64E	$31.31_{\pm 1.35}$	$56.39_{\pm 0.87}$	$50.59_{\pm 0.50}$	$35.82_{\pm 0.67}$	$35.36_{\pm 0.48}$	$71.55_{\pm 0.77}$	$31.29_{\pm 0.66}$	$41.30_{\pm 1.11}$	$52.96_{\pm 1.19}$	$52.11_{\pm 1.71}$	**45.82**
MoNE w/ NG-LBL 16E/64E	$30.38_{\pm 1.34}$	$60.31_{\pm 0.86}$	$49.17_{\pm 0.50}$	$34.58_{\pm 0.66}$	$34.82_{\pm 0.48}$	$70.84_{\pm 0.77}$	$30.81_{\pm 0.66}$	$40.94_{\pm 1.11}$	$51.38_{\pm 1.19}$	$50.70_{\pm 1.72}$	**45.36**
2.81B Activated 0.55B											
Traditional MoE 4E/64E (2.81B)	$38.65_{\pm 1.42}$	$57.89_{\pm 0.87}$	$63.00_{\pm 0.48}$	$44.38_{\pm 0.69}$	$31.61_{\pm 0.47}$	$75.19_{\pm 0.74}$	$34.74_{\pm 0.68}$	$42.37_{\pm 1.12}$	$59.98_{\pm 1.17}$	$47.89_{\pm 1.71}$	50.18
MoNE w/o NG-LBL 8E/64E (2.81B)	$39.93_{\pm 1.43}$	$61.13_{\pm 0.86}$	$63.87_{\pm 0.48}$	$46.26_{\pm 0.69}$	$38.67_{\pm 0.49}$	$76.12_{\pm 0.73}$	$34.70_{\pm 0.68}$	$42.82_{\pm 1.12}$	$59.19_{\pm 1.17}$	$43.66_{\pm 1.70}$	**50.44**
MoNE w/ NG-LBL 8E/64E (2.81B)	$37.54_{\pm 1.41}$	$62.97_{\pm 0.85}$	$63.36_{\pm 0.48}$	$46.13_{\pm 0.69}$	$36.28_{\pm 0.49}$	$76.17_{\pm 0.73}$	$34.83_{\pm 0.68}$	$42.32_{\pm 1.12}$	$59.67_{\pm 1.17}$	$57.75_{\pm 1.70}$	**51.72**

extra communication latency can be negligible. Empirically, pretraining with 50% of the activation parameters by MoNE already matches the performance of a traditional MoE, which demonstrate MoNE can effectively improve the ultilization of activated parameters.

3.3 Neuron Granular Load Balance Loss

Since We decompose each expert into neuron level sub-experts, we further introduce the *neuron granular load balance loss* (NG-LBL). NG-LBL is designed to avoid cases where a subset of neurons are rarely activated, thereby further improving parameter utilization. The formulation of NG-LBL of is similar with $\mathcal{L}_{\text{aux}}$, but is applied independently to the neuron experts within each expert.

For expert i, the fraction of tokens in the batch $\mathcal{B}$ (with batchsize of T) that assigned to neuron k, and the average gating weights that the gate projection assigned to neuron k is calculated as follows:

$$\tilde{\mathbf{f}}_{i,k} = \frac{1}{\mathrm{T}} \sum_{\mathbf{x} \in \mathcal{B}} \mathbb{1}\{k \in \texttt{argtopK}(\texttt{Abs}(\mathbf{G}_i))\}, \quad \tilde{\mathbf{P}}_{i,k} = \frac{1}{\mathrm{T}} \sum_{\mathbf{x} \in \mathcal{B}} \frac{\texttt{Abs}(\mathbf{G}_i[k])}{\sum_{t=1}^{d_{\text{expert}}} \texttt{Abs}(\mathbf{G}_i[t])}. \tag{13}$$

The whole auxiliary load balance loss used in MoNE is to sum the orginal auxiliary load balance loss and each expert's neuron granular load balance loss:

$$\tilde{\mathcal{L}}_{\text{aux}} = \mathcal{L}_{\text{aux}} + \sum_{i=1}^{\mathrm{N}_E} \mathcal{L}^i_{\text{NG-LBL}}, \quad \text{where} \quad \mathcal{L}^i_{\text{NG-LBL}} = \alpha_{NG} \cdot d_{\text{model}} \cdot \sum_{k=1}^{d_{\text{model}}} \mathbf{f}_{i,k} \cdot \mathbf{P}_{i,k} \tag{14}$$

where the $\alpha_{\text{NG-LBL}}$ is the coefficient to controls the regularization strength of NG-LBL.

4 Experiment

4.1 Experimental Setup

Model Architectures. As shown in Table 6, we implement models with total parameter counts of 925M and 2.81B. The MoE layers for both model scales contained 64 experts. Follow by Dai et al.

Table 3: Architecture exploration on different numbers of selected neurons K_N. MoNE exhibits better downstream performance when the selected rate is $1/4$.

Model	ARC-C	BOOIQ	HELLA	LAMBDA	MNLI	PIQA	RACE	SIQA	WINO	WNLI	AVG.
2.81B Activated 0.55B											
Traditional MoE 4E/64E	$38.65_{\pm 1.42}$	$57.89_{\pm 0.87}$	$63.00_{\pm 0.48}$	$44.38_{\pm 0.69}$	$31.61_{\pm 0.47}$	$75.19_{\pm 0.74}$	$34.74_{\pm 0.68}$	$42.37_{\pm 1.12}$	$59.98_{\pm 1.17}$	$47.89_{\pm 1.71}$	49.57
$K_N = 1/2 \cdot d_{\text{model}}$											
MoNE w/o NG-LBL 6E/64E	$41.04_{\pm 1.44}$	$57.34_{\pm 0.87}$	$63.50_{\pm 0.48}$	$46.21_{\pm 0.69}$	$36.98_{\pm 0.49}$	$75.68_{\pm 0.73}$	$35.12_{\pm 0.68}$	$43.35_{\pm 1.12}$	$59.76_{\pm 1.17}$	$45.07_{\pm 1.71}$	50.41
MoNE w/ NG-LBL 6E/64E	$39.59_{\pm 1.43}$	$61.96_{\pm 0.85}$	$63.10_{\pm 0.48}$	$44.89_{\pm 0.69}$	$38.41_{\pm 0.49}$	$75.30_{\pm 0.73}$	$35.22_{\pm 0.68}$	$42.17_{\pm 1.12}$	$59.12_{\pm 1.17}$	$45.07_{\pm 1.71}$	**50.48**
$K_N = 1/4 \cdot d_{\text{model}}$											
MoNE w/o NG-LBL 8E/64E	$39.93_{\pm 1.43}$	$61.13_{\pm 0.86}$	$63.87_{\pm 0.48}$	$46.26_{\pm 0.69}$	$38.67_{\pm 0.49}$	$76.12_{\pm 0.73}$	$34.70_{\pm 0.68}$	$42.82_{\pm 1.12}$	$59.19_{\pm 1.17}$	$43.66_{\pm 1.70}$	50.64
MoNE w/ NG-LBL 8E/64E	$37.54_{\pm 1.41}$	$62.97_{\pm 0.85}$	$63.36_{\pm 0.48}$	$46.13_{\pm 0.69}$	$36.28_{\pm 0.49}$	$76.17_{\pm 0.73}$	$34.83_{\pm 0.68}$	$42.32_{\pm 1.12}$	$59.67_{\pm 1.17}$	$57.75_{\pm 1.70}$	**51.70**
$K_N = 1/10 \cdot d_{\text{model}}$											
MoNE w/o NG-LBL 10E/64E	$40.27_{\pm 1.43}$	$60.86_{\pm 0.86}$	$63.89_{\pm 0.48}$	$46.09_{\pm 0.69}$	$31.59_{\pm 0.47}$	$75.84_{\pm 0.73}$	$34.55_{\pm 0.68}$	$43.09_{\pm 1.12}$	$58.01_{\pm 1.17}$	$45.07_{\pm 1.71}$	49.93
MoNE w/ NG-LBL 10E/64E	$39.85_{\pm 1.43}$	$58.10_{\pm 0.87}$	$61.99_{\pm 0.48}$	$44.21_{\pm 0.69}$	$32.07_{\pm 0.47}$	$75.03_{\pm 0.74}$	$35.12_{\pm 0.68}$	$42.12_{\pm 1.12}$	$59.27_{\pm 1.17}$	$57.75_{\pm 1.70}$	**50.55**

Table 4: The performance of MoNE when applying different activation functions. MoNE exhibits better downstream performance when applying `SiLU` and `Sigmoid`.

Model	ARC-C	BOOIQ	HELLA	LAMBDA	MNLI	PIQA	RACE	SIQA	WINO	WNLI	AVG.
`SiLU`											
Traditional MoE 4E/64E	$30.55_{\pm 1.35}$	$56.94_{\pm 0.87}$	$47.78_{\pm 0.50}$	$32.70_{\pm 0.65}$	$34.39_{\pm 0.48}$	$69.53_{\pm 0.78}$	$30.33_{\pm 0.65}$	$39.87_{\pm 1.11}$	$52.80_{\pm 1.19}$	$40.85_{\pm 1.69}$	43.38
MoNE w/o NG-LBL 8E/64E	$30.97_{\pm 1.35}$	$55.75_{\pm 0.87}$	$48.01_{\pm 0.50}$	$33.34_{\pm 0.66}$	$34.44_{\pm 0.48}$	$70.67_{\pm 0.78}$	$29.86_{\pm 0.65}$	$38.89_{\pm 1.10}$	$53.83_{\pm 1.19}$	$49.30_{\pm 1.72}$	44.67
MoNE w/ NG-LBL 8E/64E	$30.38_{\pm 1.34}$	$59.45_{\pm 0.86}$	$49.51_{\pm 0.50}$	$33.96_{\pm 0.66}$	$34.79_{\pm 0.48}$	$70.89_{\pm 0.77}$	$30.24_{\pm 0.65}$	$39.76_{\pm 1.11}$	$53.67_{\pm 1.19}$	$52.11_{\pm 1.71}$	**45.48**
`Sigmoid`											
Traditional MoE 4E/64E	$30.38_{\pm 1.35}$	$54.79_{\pm 0.87}$	$45.78_{\pm 0.50}$	$33.28_{\pm 0.65}$	$34.44_{\pm 0.48}$	$69.97_{\pm 0.78}$	$30.24_{\pm 0.65}$	$38.87_{\pm 1.11}$	$53.10_{\pm 1.19}$	$43.66_{\pm 1.70}$	43.15
MoNE w/o NG-LBL 8E/64E	$31.40_{\pm 1.36}$	$49.45_{\pm 0.88}$	$48.93_{\pm 0.50}$	$34.76_{\pm 0.66}$	$35.00_{\pm 0.48}$	$71.33_{\pm 0.77}$	$32.73_{\pm 0.67}$	$39.51_{\pm 1.11}$	$49.57_{\pm 1.19}$	$50.70_{\pm 1.72}$	44.34
MoNE w/ NG-LBL 8E/64E (Sigmoid)	$30.72_{\pm 1.35}$	$59.79_{\pm 0.86}$	$47.51_{\pm 0.50}$	$33.75_{\pm 0.66}$	$34.80_{\pm 0.48}$	$70.18_{\pm 0.78}$	$32.82_{\pm 0.67}$	$40.17_{\pm 1.11}$	$51.38_{\pm 1.19}$	$53.52_{\pm 1.71}$	**45.46**
`Softmax`											
Traditional MoE 4E/64E	$27.93_{\pm 1.31}$	$48.20_{\pm 0.88}$	$36.67_{\pm 0.48}$	$29.82_{\pm 0.64}$	$33.28_{\pm 0.48}$	$66.51_{\pm 0.80}$	$29.96_{\pm 0.65}$	$37.84_{\pm 1.10}$	$49.28_{\pm 1.19}$	$40.66_{\pm 1.69}$	40.02
MoNE w/o NG-LBL 8E/64E	$28.24_{\pm 1.31}$	$47.52_{\pm 0.88}$	$38.30_{\pm 0.49}$	$30.08_{\pm 0.64}$	$35.00_{\pm 0.48}$	$66.43_{\pm 0.80}$	$30.33_{\pm 0.65}$	$39.25_{\pm 1.10}$	$51.54_{\pm 1.19}$	$42.25_{\pm 1.70}$	40.96
MoNE w/ NG-LBL 8E/64E	$28.58_{\pm 1.32}$	$48.53_{\pm 0.88}$	$44.96_{\pm 0.50}$	$32.52_{\pm 0.65}$	$35.16_{\pm 0.48}$	$69.91_{\pm 0.78}$	$30.43_{\pm 0.66}$	$38.89_{\pm 1.10}$	$50.59_{\pm 1.19}$	$46.48_{\pm 1.71}$	**41.81**

(2024), we use one shared expert. The MoE layers in both model scales comprise 64 experts. For the smaller model, we trained traditional MoE with 4, 6 and 8 experts-per-token, which correspond to activated parameters of 290M, 310M and 330M, respectively; For the larger model, we trained a traditional MoE with 4 experts-per-token, corresponding to 0.55B activated parameters. In the main experimental suite, each traditional MoE configuration was paired with a corresponding MoNE: the K_N is set to $d_{\text{model}}/4$ and K_E is set two times of corresponding traditional MoE. so that the number of activated parameters is equal between the traditional MoE and MoNE. Please refer Appendix A.1 for detailed training hyper-parameters.

Data & Tokenizer. We trained the 925M parameter model on a 50B-token subset of the NeMaTron-CC dataset (Su et al., 2024) and the 2.81B parameter model on a 100B-token subset of the NeMaTron-CC dataset. All text was tokenized with the LLaMA-3-8B tokenizer (Dubey et al., 2024).

Hyper-Parameters. The hyper-parameters are selected based on the common practice for dense language models. We replace all FFN layers with MoE layer in the transformer. Please refer Appendix A.1 for detailed training hyper-parameters.

Benchmarks. We use the lm-evaluation-harness (Gao et al., 2024) for evaluation. The benchmarks used include ARC-C (Clark et al., 2018), BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), LAMBADA (Paperno et al., 2016), MNLI (Williams et al., 2017), PIQA (Bisk et al., 2020), RACE (Lai et al., 2017), SIQA (Sap et al., 2019), WinoGrande (Sakaguchi et al., 2021), WNLI (Levesque et al., 2012). For all these benchmarks, we report the zero-shot accuracy.

4.2 Main Results

Prior Experiment We first compare traditional MoE and MoNE with the same number of experts per token N_E. For MoNE, we set the number of used neurons K_N to $d_{\text{model}}/4$, which reduces the number of MoNE's activated parameters in the moe layer to approximately half of the traditional MoE's. As an additional baseline, we select a random subset of size $K_N = d_{\text{model}}/4$ from the neuron experts instead of the `TopK` strategy to pretrain the model. Table 1 shows that MoNE attains performance comparable to the traditional MoE while using 50% of the activated parameters in the

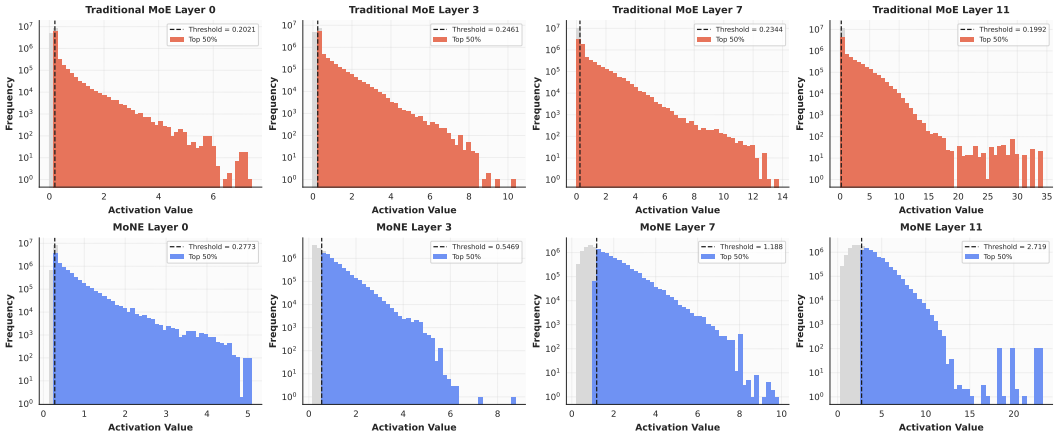

Figure 4: The comparison of the activation value $\mathbf{G}$ for the neuron experts between traditional MoE and MoNE. MoNE effectively increase the activation weight compared with traditional MoE.

MoE layer, whereas the random subset baseline suffers a large drop in performance. These findings indicate that MoNE's selection mechanism effectively improve the utilization of the activated parameters.

Comparisons to Traditional MoE of Equivalent Activated Parameters We further compared MoNE with Traditional MoE with equivalent activated parameters. As shown in the Table 2, MoNE consistently improves over the traditional MoE, and the improvement grows as the number of activated parameters increases. Concretely, when activating 290M parameters MoNE yields an improvement of approximately 1% relative to the traditional MoE, and this gain increases to about 2% at 330M activated parameters. Also, with 330M activated parameters, MoNE attains performance that is comparable to a dense model with 925M parameters. For the 3B models MoNE improves upon the traditional MoE by roughly 1.1%. These results indicate that MoNE is a promising approach for training MoE-like models.

4.3 Further Analysis

The Effectiveness of NG-LBL We investigate the effect of NG-LBL on MoNE in Table 2 and Table 3. Empirically, NG-LBL consistently improves MoNE's performance and increases its parameter efficiency: on tue 1B-parameter model, NG-LBL yields an improvement about 1.0%, while on a 3B-parameter model the gain is about 1.4%. Figure 5 shows that NG-LBL substantially accelerates the decline of training loss,which demonstrates the effectiveness of NG-LBL. To better understand how NG-LBL helps, we examine load balancing among neuron experts. As shown in Figure 6, neuron experts achieve better load balance with NG-LBL. This indicate that the balancing at the neuron level can effectively increase the ability of experts

The Analysis of the Activation Weight Figure 4 compares the activation weight of traditional MoE and MoNE. MoNE effectively increases neuron activation weight: the median value of activation weight in the first layer rises from 0.20 to 0.28 and continues to grow with depth. The median value increases from 0.20 to 2.70 in the final layer. In addition, the activation weight's distribution produced by MoNE is noticeably more uniform, indicating a more balanced utilization of neuron experts. These results demonstrate that MoNE both increases and homogenizes the use of neuron experts, thereby reducing the sparsity of activated parameters and improving the utilization of the activated parameters.

The Effect of the Neuron Expert Activation Ratio Figure 1 indicates that the within-expert activation rate has a meaningful effect. Keeping the number of activated parameters fixed, we pretrained models with different K_N; Table 3 shows that all settings improve over the baseline, with the best performance at a ratio of $1/4$. Therefore, we recommend using $\mathrm{K}_N = 1/4 \cdot d_{\mathrm{model}}$. The result aligns with the result in Figure 1: model performance remains stable until approximately 70% of neurons

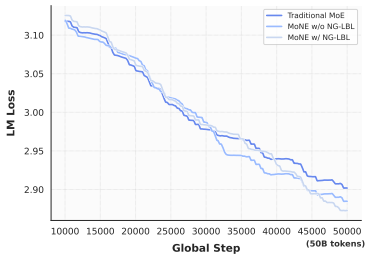

Figure 5: Pre-training loss between traditional MoE and MoNE

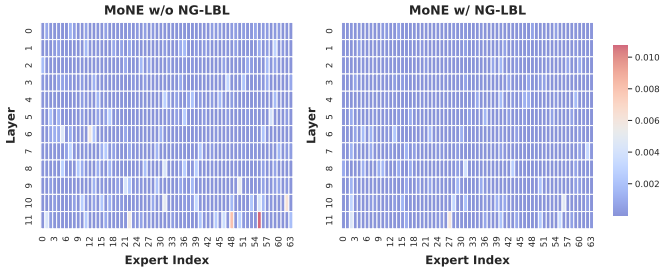

Figure 6: The visualization of load balance for different layers and experts, the value visualized is the variance of $\tilde{\mathbf{f}}_{i,k}$.

are removed, which implies that each activated expert only needs 30% of its neurons to process an input.

The Effect of Different Activation Function inside the Expert We further explored three internal activation functions for neuron experts in MoNE—`Sigmoid`, `SiLU`, and `Softmax`. Empirically, `Sigmoid` and `SiLU` produce consistently performance than `Softmax`. While experts contain large amounts of neurons, `Softmax` concentrates probability distribution on a small subset while assigning near-zero weights to most neurons, thereby reducing effective parameter usage. Using NG-LBL mitigates this concentration by encouraging more uniform neuron activation, but `Softmax` still underperforms the baseline in our experiments. Accordingly, we recommend use `Sigmoid` or `SiLU` as the default internal activation for MoNE architectures.

Table 5: Throughput and memory usage comparison among traditional MoE and MoNE. The model with 925M parameters and 290M activation parameters is used for calculation. Auxiliary losses do not impact efficiency.

	Traditional MoE	MoNE
Configuration		
Batch size	8	8
Input length	1024	1024
New tokens	128	128
Throughput & Memory		
Tokens/sec	1340.63	1338.49
Memory Peak Reserved	7.8GB	7.8GB

The Efficiency of MoNE We further investigate the efficiency of MoNE. As shown in Table 5, with the same number of activated parameters, MoNE and traditional MoE require comparable GPU memory and show nearly identical throughput. Crucially, MoNE achieves neuron granular expert selection without enlarging the router or increasing communication latency overhead. Hence, MoNE provides a practical approach to neuron granular expert computation while preserving computation efficiency.

5 Conclusion

In this work, we demonstrate that the parameters activated by Mixture-of-Experts (MoE) layers is also highly sparse. By decomposing each expert into neuron granular subexperts. we find that many neuron experts receive very small activation weights. The result motivate us to improve the utilization of activated paratmeters by only use the neuron experts with high activation weights. Therefore We propose Mixture of Neuron Experts (MoNE), a simple and practical modification of traditional MoE that operates at neuron granularity: by decomposing experts into neuron granular subexperts and applying a simple sorting operation to the gate-projection outputs prior to expert computation, MoNE converts a traditional MoE into a neuron granular MoE. Furthermore, we propose to apply the neuron granular load-balance loss on the neuron experts to encourage more uniform neuron utilization. MoNE requires no additional model parameters and incurs only a negligible computational overhead relative to traditional MoE. Empirically, MoNE matches baseline performance while activating only half of the parameters in the MoE layer and achieves consistent improvements when compared at equal numbers of activated parameters. Expert granularity is an important focus of current MoE development, while traditional MoE faces problems such as large routers and large

communication delays when expert partitioning is overly fine granularity,. We believe MoNE is a practical step toward more efficient and scalable MoE-like architectures.

ETHICS STATEMENT

This paper presents work whose goal is to advance the field of large language model. There are many potential consequences of our work, none of which we feel must be specifically highlighted here.

REPRODUCIBILITY STATEMENT

The details of datasets, model architectures and hyper-parameters are described in Section 4.1 and Appendix A.1.

REFERENCES

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

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

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

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

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

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*, 2019.

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

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

Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In *International conference on machine learning*, pp. 933–941. PMLR, 2017.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, et al. Glam: Efficient scaling of language models with mixture-of-experts. In *International conference on machine learning*, pp. 5547–5569. PMLR, 2022.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pp. arXiv–2407, 2024.

William Fedus, Jeff Dean, and Barret Zoph. A review of sparse expert models in deep learning. *arXiv preprint arXiv:2209.01667*, 2022a.

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

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

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M Roy, and Michael Carbin. Stabilizing the lottery ticket hypothesis. *arXiv preprint arXiv:1903.01611*, 2019.

Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International conference on machine learning*, pp. 10323–10337. PMLR, 2023.

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, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. URL `https://zenodo.org/records/12608602`.

Xinyang Geng and Hao Liu. Openllama: An open reproduction of llama, May 2023. URL `https://github.com/openlm-research/open_llama`.

Xu Owen He. Mixture of a million experts. *arXiv preprint arXiv:2407.04153*, 2024.

Zihao Huang, Qiyang Min, Hongzhi Huang, Defa Zhu, Yutao Zeng, Ran Guo, and Xun Zhou. Ultra-sparse memory network. *arXiv preprint arXiv:2411.12364*, 2024.

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

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

Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. Race: Large-scale reading comprehension dataset from examinations. *arXiv preprint arXiv:1704.04683*, 2017.

Guillaume Lample, Alexandre Sablayrolles, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Large memory layers with product keys. *Advances in Neural Information Processing Systems*, 32, 2019.

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

Hector J Levesque, Ernest Davis, and Leora Morgenstern. The winograd schema challenge. *KR*, 2012(13th):3, 2012.

Zichong Li, Chen Liang, Zixuan Zhang, Ilgee Hong, Young Jin Kim, Weizhu Chen, and Tuo Zhao. Slimmoe: Structured compression of large moe models via expert slimming and distillation. *arXiv preprint arXiv:2506.18349*, 2025.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Saeed Masoudnia and Reza Ebrahimpour. Mixture of experts: a literature survey. *Artificial Intelligence Review*, 42(2):275–293, 2014.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The lambada dataset: Word prediction requiring a broad discourse context. *arXiv preprint arXiv:1606.06031*, 2016.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

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

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. Socialiqa: Commonsense reasoning about social interactions. *arXiv preprint arXiv:1904.09728*, 2019.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. BLOOM: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*, 2022.

N Shazeer, A Mirhoseini, K Maziarz, A Davis, Q Le, G Hinton, and J Dean. The sparsely-gated mixture-of-experts layer. *Outrageously large neural networks*, 2, 2017.

Dan Su, Kezhi Kong, Ying Lin, Joseph Jennings, Brandon Norick, Markus Kliegl, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. Nemotron-cc: Transforming common crawl into a refined long-horizon pretraining dataset. *arXiv preprint arXiv:2412.02595*, 2024.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.

Qwen Team. Qwen3 technical report, 2025.

Changxin Tian, Kunlong Chen, Jia Liu, Ziqi Liu, Zhiqiang Zhang, and Jun Zhou. Towards greater leverage: Scaling laws for efficient mixture-of-experts language models. *arXiv preprint arXiv:2507.17702*, 2025.

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

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

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

Adina Williams, Nikita Nangia, and Samuel R Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*, 2017.

Haoyuan Wu, Haoxing Chen, Xiaodong Chen, Zhanchao Zhou, Tieyuan Chen, Yihong Zhuang, Guoshan Lu, Zenan Huang, Junbo Zhao, Lin Liu, et al. Grove moe: Towards efficient and superior moe llms with adjugate experts. *arXiv preprint arXiv:2508.07785*, 2025.

Fuzhao Xue, Zian Zheng, Yao Fu, Jinjie Ni, Zangwei Zheng, Wangchunshu Zhou, and Yang You. Openmoe: An early effort on open mixture-of-experts language models. *arXiv preprint arXiv:2402.01739*, 2024.

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

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. Tinyllama: An open-source small language model. *arXiv preprint arXiv:2401.02385*, 2024.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

A APPENDIX

A.1 EXPERIMENTAL DATAILS

A.1.1 MODEL ARCHITECTURES.

We list the model configuration in Table 6. Here we verified the corresponding MoE and MoNE has the same number of activated parameters. Suppose the number of parameters for `gate projection`, `up projection` and `down projection` is N. For a MoE layer with 4 experts activated, the number of activated parameters is $4 \cdot 3 \cdot \mathrm{N}$. For a MoNE layer with 6 experts activated and N_k /d_{model} is $1/2$, the number of activated parameters of an expert can be calculated as $\mathrm{N}+1/2 \cdot 2 \cdot \mathrm{N}$, where the parameters of `gate projection` are all activated, and the the parameters of `up projection` and `down projection` only activated $1/2$. Then total activated parameters can be calculated as $6 \cdot (\mathrm{N} + 1/2 \cdot 2 \cdot \mathrm{N}) = 12\mathrm{N}$. Accordingly, for a MoNE layer with 8 experts activated and N_k /d_{model} is $1/4$, the number of activated parameters is $8 \cdot (\mathrm{N} + 1/4 \cdot 2 \cdot \mathrm{N}) = 12\mathrm{N}$, and for a MoNE layer with 10 experts activated and N_k /d_{model} is $1/10$, the number of activated parameters is $10 \cdot (\mathrm{N} + 1/10 \cdot 2 \cdot \mathrm{N}) = 12\mathrm{N}$. Therefore the corresponding MoE and MoNE has the same number of activated parameters.

Table 6: **Sizes and architectures of MoNE and traditional MoE models.** "290M/925M" represents an architecture of an approximately 925M parameter, with 290M activated per token during inference.

Methods	# Layers	# Hidden Size	# Intermediate Size	# Heads	# Head Dim	# The Number of FFN Experts	# The Number of Experts per Token	# $\mathrm{N}_k/d_{\mathrm{model}}$
Traditional MoE 290M/925M	12	768	368	16	48	64	4	-
Traditional MoE 310M/925M	12	768	368	16	48	64	6	-
Traditional MoE 330M/925M	12	768	368	16	48	64	8	-
Traditional MoE 0.55B/2.81B	24	1024	512	16	96	64	4	-
MoNE 290M/925M	12	768	368	16	48	64	8	1/4
MoNE 310M/925M	12	768	368	16	48	64	12	1/4
MoNE 330M/925M	12	768	368	16	48	64	16	1/4
MoNE 0.55B/2.81B 6E	24	1024	512	16	96	64	6	1/2
MoNE 0.55B/2.81B 8E	24	1024	512	16	96	64	8	1/4
MoNE 0.55B/2.81B 10E	24	1024	512	16	96	64	10	1/10

A.1.2 HYPER-PARAMETERS.

The hyperparameters are selected based on the common practice for dense transformer language models (Zhang et al., 2024; Geng & Liu, 2023; Touvron et al., 2023b; Xue et al., 2024). The key training hyperparameters used in our experiments are as follows: batch size (tokens) $= 1\,\mathrm{M}$; auxiliary load-balance weight $\alpha_{\text{aux}} = 0.001$; neuron-granular load-balance weight $\alpha_{\mathrm{NG}} = 0.001$; optimizer = FusedAdam (Kingma, 2014); learning rate $= 5e-4$; router scoring activation function = softmax; weight decay (Loshchilov & Hutter, 2017) $= 0.1$; model maximum sequence length $= 2\mathrm{k}$. These settings were kept fixed across the reported pretraining runs and ablations unless stated otherwise.

A.1.3 CALCULATE RESOURCES AND ENVIRONMENT

We use deepspeed as the training framework. For the 925M model, We conduct training on a cluster with 4 nodes and 32 A100 GPUs. For the 2.81B model, We conduct training on a cluster with 16 nodes and 128 A100 GPUs.

A.2 Additional Experiment

A.2.1 More results on the activation value for the neuron experts.

We visualize additional neuron granular activation values for Qwen3-30B-A3B and DeepSeek-V2-Lite. As shown in Figure 7 and Figure 8, the vast majority of neuron experts receive negligible gate weights: most entries of $\mathbf{G}$ are close to zero, indicating that a large fraction of neurons within each expert remain effectively inactive during inference.

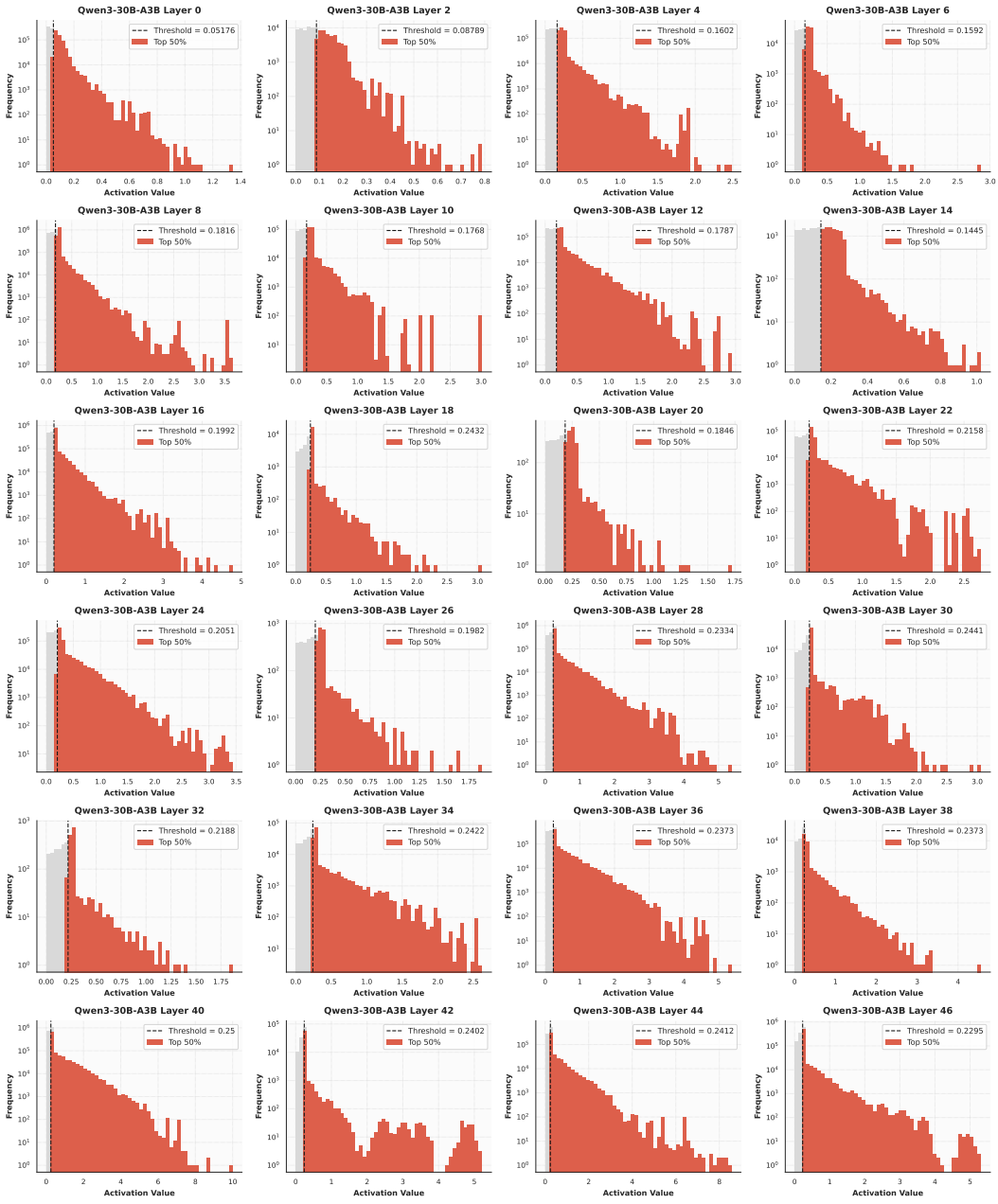

Figure 7: The activation value for the neuron experts on Qwen3-30B-A3

We further compares the activation weight of traditional MoE and MoNE.As shown in Figure 9 and Figure 10 MoNE effectively increases neuron activation weight: the median value of activation weight in the first layer rises from 0.20 to 0.28 and continues to grow with depth. The median

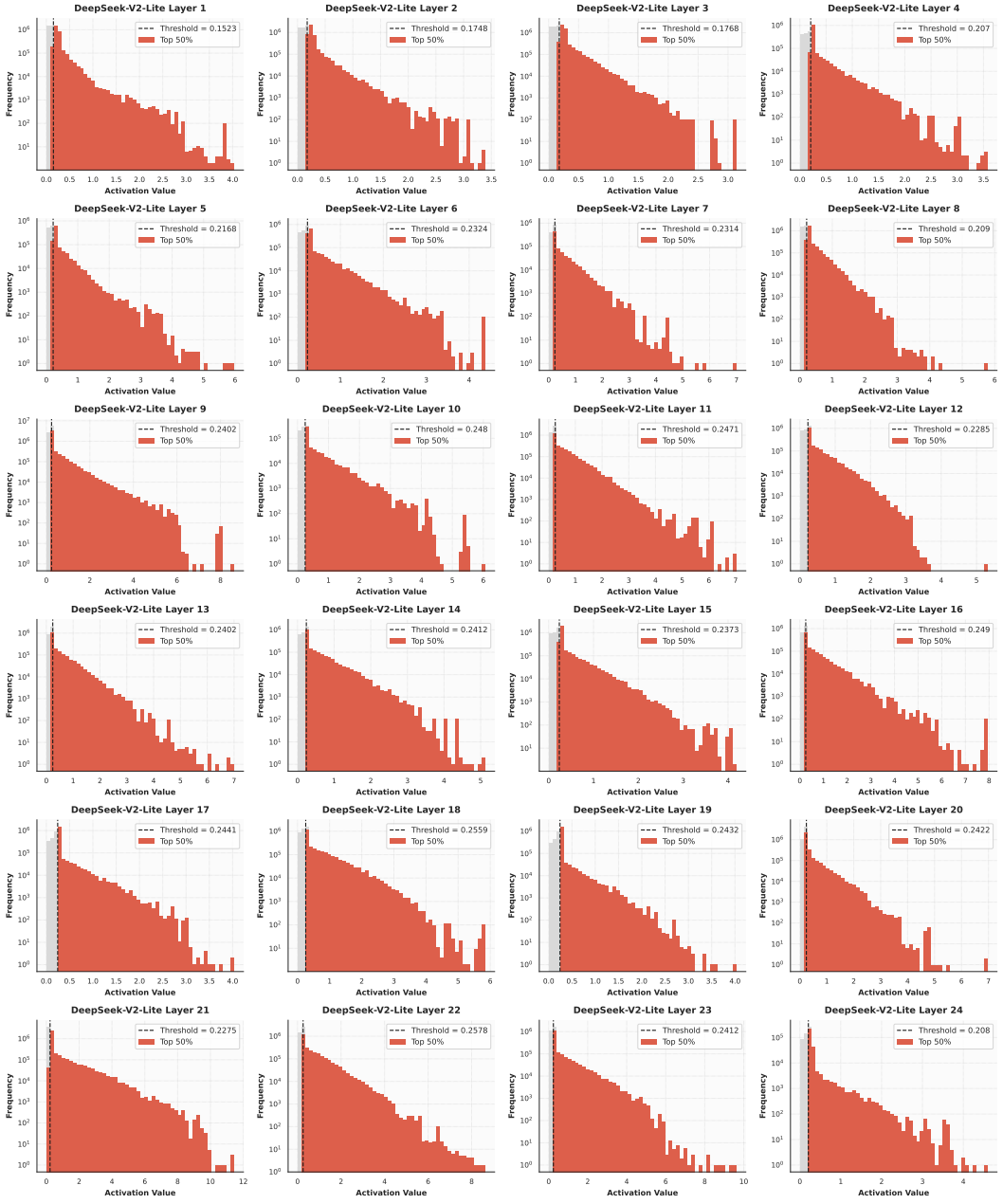

Figure 8: The activation value for the neuron experts on DeepSeek-V2-Lite

value increases from 0.20 to 2.70 in the final layer. In addition, the activation weight's distribution produced by MoNE is noticeably more uniform, indicating a more balanced utilization of neuron experts. These results demonstrate that MoNE both increases and homogenizes the use of neuron experts, thereby reducing the sparsity of activated parameters and improving the utilization of the activated parameters.

A.2.2 The comparison of traning loss for traditional MoE and MoNE

As shown in Figure 11, MoNE exhibit more effective expert learning compared with traditional MoE, as evidenced by lower loss values.

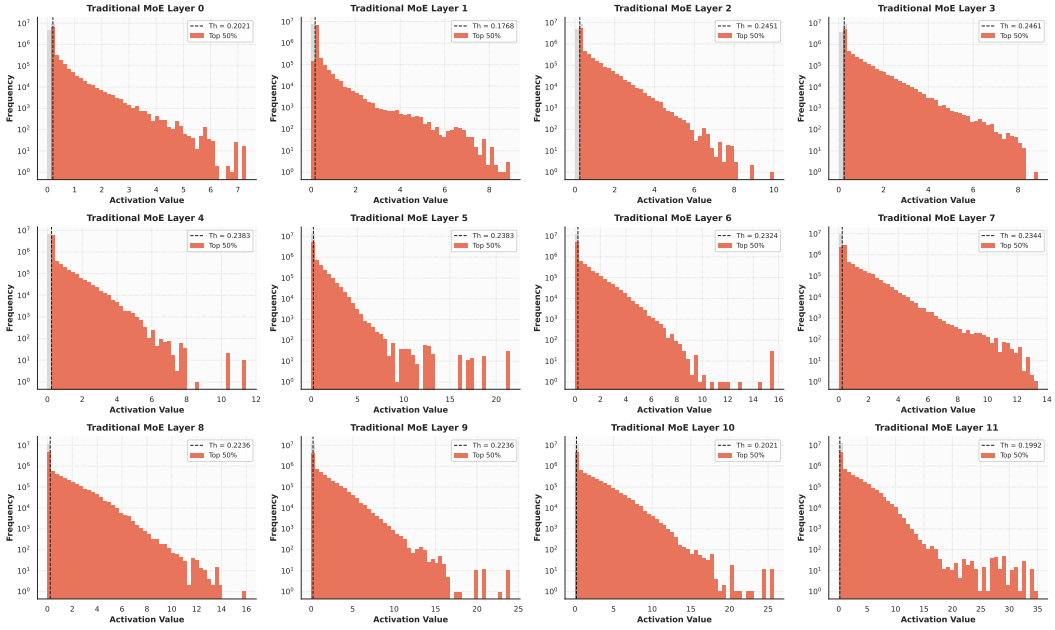

Figure 9: The activation value for the neuron experts on Traditional MoE

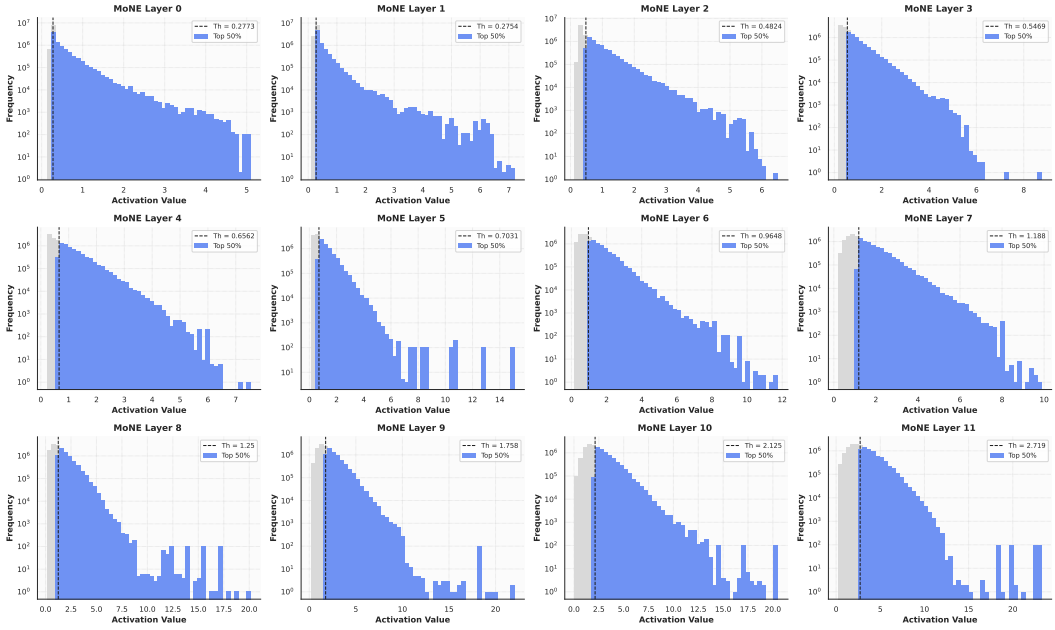

Figure 10: The activation value for the neuron experts on MoNE

A.2.3 THE EFFECT OF $\mathcal{L}_{\text{AUX}}$ ON MONE

We further investigate the influence of an auxiliary load-balance loss $\mathcal{L}_{\text{aux}}$ on MoNE. Our experimental results show that $\mathcal{L}_{\text{aux}}$ significantly affects MoNE's performance, suggesting that the balacnce across experts is important for MoNE to realize further gains in parameter utilization.

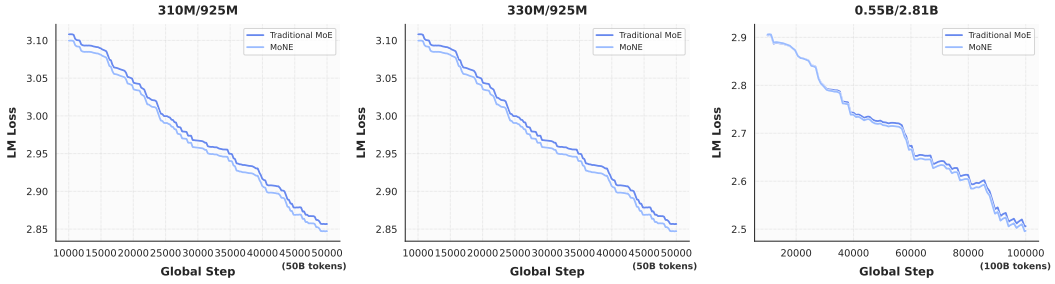

Figure 11: Pre-training loss between tradition MoE and MoNE.

Table 7: The ablation study on auxiliary load-balance loss $\mathcal{L}_{\text{aux}}$

Model	ARC-C	BOOIQ	HELLA	LAMBDA	MNLI	PIQA	RACE	SIQA	WINO	WNLI	AVG.
925M Activated 290M											
Traditional MoE w/o $\mathcal{L}_{\text{aux}}$ 4E/64E	28.33	46.85	45.63	33.44	34.58	68.88	30.72	38.69	52.49	50.70	43.03
Traditional MoE w/ $\mathcal{L}_{\text{aux}}$ 4E/64E	30.55	56.94	47.78	32.70	34.39	69.53	30.33	39.87	52.80	40.85	43.57
MoNE w/o $\mathcal{L}_{\text{aux}}$ 8E/64E	28.67	52.72	44.93	32.06	34.85	69.48	30.14	39.92	53.91	42.25	42.89
MoNE w/ $\mathcal{L}_{\text{aux}}$ 8E/64E	30.97	55.75	48.01	33.34	34.44	70.67	29.86	38.89	53.83	49.30	44.51

A.3 LLM USAGE

This study utilizes Large Language Models (LLMs) to refine content, adjust formatting, construct tables, and provide writing suggestions for specific chapters.