### LIBMOE: A LIBRARY FOR COMPREHENSIVE BENCHMARKING MIXTURE OF EXPERTS IN LARGE LANGUAGE MODELS

### Nam V. Nguyen 1 Thong T. Doan 1 Luong Tran 1 Van Nguyen 1 Quang Pham 12 ABSTRACT

Mixture of Experts (MoEs) plays an important role in the development of more efficient and effective large language models (LLMs). Due to the enormous resource requirements, studying large scale MoE algorithms remain in-accessible to many researchers. This work develops LibMoE, a comprehensive and modular framework to streamline the research, training, and evaluation of MoE algorithms. Built upon three core principles: (i) modular design, (ii) efficient training; (iii) comprehensive evaluation, LibMoE brings MoE in LLMs more accessible to a wide range of researchers by standardizing the training and evaluation pipelines. Using LibMoE, we extensively benchmarked five state-of-the-art MoE algorithms over three different LLMs and 11 datasets under the zero-shot setting. The results show that despite the unique characteristics, all MoE algorithms perform roughly similar when averaged across a wide range of tasks. With the modular design and extensive evaluation, we believe LibMoE will be invaluable for researchers to make meaningful progress towards the next generation of MoE and LLMs. Project page: https://fsoft-aic.github.io/fsoft-LibMoE.github.io.

#### 1 Introduction

Recent years have witnessed a tremendous success of Mixture-of-Experts (MoE) and its sparse variant (MoE) in facilitating the training of large language models (LLMs) (Shazeer et al., 2017; Lepikhin et al., 2020; Fedus et al., 2022; Dai et al., 2024; Jiang et al., 2024; Abdin et al., 2024) and multimodal LLMs (Du et al., 2022; Chen et al., 2024b; Xue et al., 2024; Li et al., 2024b; Han et al., 2024). By only activating a subset of parameters per input, MoE greatly improves the training efficiency, and allows researchers to train models with hundreds of billions parameters with low computational overhead. Interestingly, MoE algorithms often achieved superior performances compared to their dense counterpart (Shazeer et al., 2017), which has gather an increased interest in the community to develop more advanced algorithms and better theories to understand their behaviour (Xue et al., 2024; Chen et al., 2024a). Thus, MoE and SMoE have becoming one of the most prominent strategy towards the next generation of intelligent machines.

Despite the algorithmic advancements, training MoE models is undeniably costly, making large-scale studies accessible only to a few groups of researchers with an enormous amount of compute resources. For example, Muennighoff et al. (2024) needed 256 H100 GPUs to train OLMOE-1B-7B, and Lin et al. (2024) used 256 A100 GPUs for their experiments. On the other hand, a majority of researchers

<quangp2808@gmail.com>.

can only afford to test their ideas on small settings (Chi et al., 2022; Do et al., 2023; Pham et al., 2024; Nguyen et al., 2024a), or even synthetic datasets (Nguyen et al., 2024b;c). Such a discrepancy severely undermines the potential of MoE where it truly shines under the large scale training settings. Consequently, the most successful models in practice (Wei et al., 2024; Sukhbaatar et al., 2024; Xie et al., 2022; Dai et al., 2022) are still based on the original variant proposed in (Fedus et al., 2022) although there exists a plethora of advanced MoE algorithms (Zhou et al., 2022; Huang et al., 2024; Do et al., 2023; Pham et al., 2024), they introduce significant changes in the operational methods and gating mechanisms of MoE.

This work aims at providing a streamlined toolkit to facilitate the research of MoE and SMoE in LLMs. To this end, we carefully develop LibMoE to support distributed training and comprehensive evaluations of various MoE algorithms. With a modular design, LibMoE provides not only a comprehensive evaluation over a wide range of zero-shot tasks, but also a full customization of MoE algorithms, including sparsity, expert-router interactions, balancing losses, etc. Furthermore, we incorporate the state-of-the-art technique of sparse upcycling (Komatsuzaki et al., 2022) to skip the expensive costs of training from scratch, allowing MoE to be incorporated into any existing dense LLM checkpoints at an affordable compute. Notably, our full training pipeline can be completed within 55 hours using only 4 × A100 GPUs while the MoE upcyling step can be finished within 32 hours only. Consequently, LibMoE offers an experiment setting that is affordable to a wide-range of researchers while the evaluation remains faithful and truly reflects the

<sup>&</sup>lt;sup>1</sup>FPT Software AI Center, Viet Nam <sup>2</sup>VNUHCM - Uni-Quang Pham versity of Science. Correspondence to:

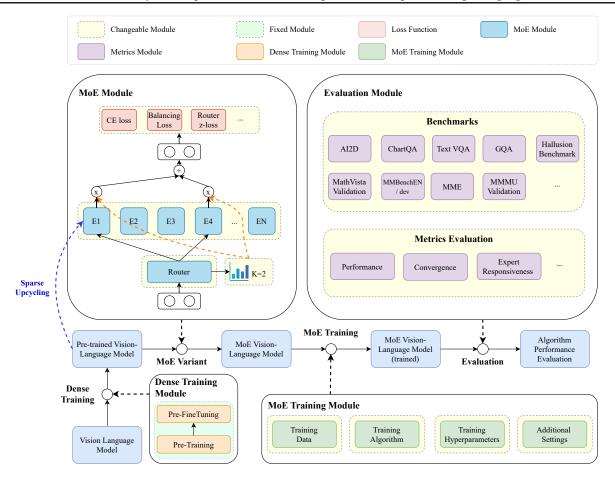


Figure 1. The detailed design of LiBMoE, which comprises three major modules. First, the MoE module implements various MoE algorithms. Second, the training modules handles the training process and supports various configurations. Lastly, the evaluation module supports almost 100 zero-shot benchmarks and a wide-range of metrics.

algorithms generalization capabilities. Figure 1 outlines the key components and design principle of LibMoE while Section 3.3 will disscuss these aspects in more details. Using LibMoE, we extensively evaluate five state-of-the-art MoE algorithms, namely: SMoE Router (Shazeer et al., 2017), Cosine Router (Chi et al., 2022), Sigmoid Router (Csordás et al., 2023), Hyper Router (Do et al., 2023), Perturbed Cosine Router (Nguyen et al., 2024a); with three model configurations. At the end of training, we calculate the model generalization over 11 zero-shot benchmarks. The results showed marginal differences in the overall performances of the algorithms considered. Moreover, a closer look at the evolution of the performance throughout training reveals that there exists a checkpoint that achieved much better results than the final one, highlighting a need for an early stopping mechanism for training. Lastly, we conducted extensive experiments to explore various MoE's characteristics such as expert assignment, expert selection, and the impacts of different vision encoders, which serve as promising research directions for future studies.

In summary, this work makes the following contributions. First, we present LibMoE, a comprehensive toolkit to streamline the development of MoE in LLMs. Second, with LibMoE, we implemented a standard benchmark to standardize the evaluation of five state-of-the-art MoE algorithms. Lastly, LibMoE facilitates research beyond reporting the final performance by allowing researchers to easily explore various factors such as early-stopping, expert assignments, architecture choices, and many more.

#### 2 RELATED WORK

#### 2.1 Mixture of Experts

MoE is an important class of machine learning algorithms and has been extensively studied in the literature (Jacobs et al., 1991). Recently, its sparse variant, SMoE (Shazeer et al., 2017), has gathered much interests in the community thanks to its successful applications in scaling up LLMs (Abdin et al., 2024; Jiang et al., 2024; Yang et al., 2024; Wei et al., 2024). Since then, extensive efforts have been devoted

to develop innovative algorithms (Du et al., 2022; Chen et al., 2024d; Li et al., 2024b), advanced training infrastructures (Xue et al., 2024), and sophisticated theories (Nguyen et al., 2024b;a;c).

Despite the encouraging progress, we wish to highlight a clear discrepancy in existing MoE studies. Particularly, despite a plethora of advanced algorithms and theories, the most advanced and capable LLMs (Jiang et al., 2024; Li et al., 2024a; Abdin et al., 2024) are still based on the original variant proposed in Fedus et al. (2022). This suggests that our current theoretical understanding remains quite limited to the academic setting and has not made a great impact to the real-world scenarios, which we believe to be the most exciting domain to study SMoE. Such a gap stems from the expensive barrier to entry such as obtaining massive datasets and computing resources, which is out of reach for most research groups. Therefore, we develop LibMoE not only to bring SMoE accessible to the majority of researchers but also to tally the state-of-the-art algorithms in a fair, comprehensive, and large-scale manner.

#### 2.2 Mixture of Experts Toolkits

MoE and SMoE can be implemented by a few popular open-sources toolkits such as FastMoE (He et al., 2021), OpenMoE+t5x (Xue et al., 2024), Tutel (Hwang et al., 2023). However, we argue that such toolkits are not ideal for most researchers to explore state-of-the-art SMoEs. Particularly, Tutel or OpenMoE+t5x only support the large scale pretraining setting over many GPUs. In contrast, FastMoE provides an outdated implementation of SMoE which does not support the recent LLMs or advanced distributed training libraries such as Deepspeed (Lian et al., 2024).

On the other hand, we design LibMoE for the majority of researchers to study MoE. Researchers can start training with only 1e9 tokens, which is 1,000 times fewer than Open-MoE. Moreover, LibMoE can load any pre-trained LLMs and incorporates advanced distributed training strategies such as data parallelism and model sharding, which are not supported by FastMoE. Lastly, LibMoE also supports evaluation of almost 100 zero-shot benchmarks, many of which can be directly compared with the most state-of-the-art models such as GPT-40 or Claude 3.5. Therefore, LibMoE not only is suitable for the majority of researchers but also scales exceptionally well with the computational resources.

#### 3 LIBMOE

#### 3.1 Preliminary: Mixture of Experts

The core methodology of a Sparse Mixture-of-Experts (MoE) is using a gating network to activate different subsets of expert networks for different inputs. An MoE layer includes a set of N experts  $(E_1, E_2, ..., E_N)$ , where each

expert is structurally identical to a standard FFN, and a gating network (G) designed to choose the top-k experts with the largest affinity scores. Building on the framework established by Shazeer et al. (2017), the mathematical representation of MoE layer can be expressed as:

$$y = \sum_{i=1}^{N} G_i(x) E_i(x),$$
 (1)

$$G(x) = \text{TopK}(\sigma(xW_q + b) + R_{\epsilon}, k), \qquad (2)$$

$$TopK(G(x), k)_i = \begin{cases} G(x)_i, & \text{if } G(x)_i \text{ is the top } k, \\ 0, & \text{otherwise.} \end{cases}$$
 (3)

Where x is the input, N denotes the total number of experts, and K denotes the number of experts activated per sample. The gate model  $G(\cdot)$  is a Noisy Top-K Network (Shazeer et al., 2017) with parameters  $W_g$  and noise  $R_\epsilon$ .  $G(x)_i$  denotes the weight value for the i-th expert, and  $\sigma(\cdot)$  denotes a similarity measure, commonly implemented as a softmax function. This sparsity character ensures the conditional computation in the MoE layer that each token be assigned and computed only k experts.

#### 3.2 Training and Evaluation Pipelines

A key challenge in training MoE models is obtaining a massive dataset and a large amount of compute. In this work, we propose to incorporate MoE training into any existing dense LLM checkpoints via the Sparse Upcycling technique (Komatsuzaki et al., 2022), which duplicates the original model to create experts and continue training them on a downstream dataset as a normal MoE. Consequently, we can bypass the expensive pre-training step and evaluate MoE algorithms with the most advanced public LLMs.

**Training pipeline** For training, we adopt the visionlanguage pre-training task, which is one of the more challenging problem and only requires a rather small amount of data to start (around 1e9 tokens). To this end, we follow the CUMO framework (Li et al., 2024b) to upcycle the LLaVA model (Liu et al., 2023a), which consists of three modules: a pre-trained visual encoder, a pre-trained LLM, and a randomly initialized MLP connector. Training follows a two-stage process. First, the dense training stage initializes the MLP connector and train it to connect the pretrained visual encoder to the pre-trained LLM. Second, the MoE training stage upcyles the model to become MoE and also trains all components to obtain the visual instruction following capabilities, Figure 2 shows the model training process in detail. Importantly, we follow Li et al. (2024b) to only upcyle the MLP connector and the visual encoder since upcycling a dense LLM is found to be worse than just using an MoE LLM. Moreover, we highlight that the dense

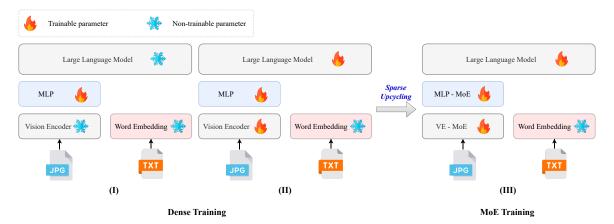


Figure 2. Overview of the LibMoE architecture and training process. In the first stage of Dense Training, only the MLP is trained to improve alignment. In the second stage, all parameters are trained. During MoE Training, the feed-forward networks (FFNs) of the Vision Encoder (VE) and MLP Connector are used to initialize the experts within the MoE framework, and all parameters continue to be trained.

training stage is unrelated to MoE, and thus the checkpoints can be re-used to train different MoE algorithms. Overall, in our small scale setting with 4xA100 GPUs, the whole training process takes a bit over 55 hours while the third training stage takes 32 hours, which is an affordable cost.

**Evaluation pipeline** Classical machine learning studies often follow the training-validation-testing split to evaluate the model on a dataset. Many existing MoE studies (Chi et al., 2022; Do et al., 2023) also have followed this framework where a different model is trained and then evaluated for different benchmarks. In contrast, LLMs have presented a paradigm shift where a single model trained on a massive amount of data can generalize to many benchmarks without seeing its training data, also referred to as the zero-shot evaluation. Thus, to drive the MoE developments towards real-world scenarios, we implement LibMoE to evaluate the algorithms in the zero-shot setting. To this end, we modify the LMMS-Eval framework (Zhang et al., 2024) to evaluate the final checkpoints of various MoE algorithms. Particularly, we carefully select 11 popular benchmarks provided by LMMS-Eval and report the evaluation results. Additionally, we also provide a LibMoE's model loader so that future users can freely explore almost 100 benchmark supported by LMMS-Eval.

#### 3.3 Design Principles

We design LibMoE to be modular and highly extensible. At a high level, LibMoE comprises three major modules. First, the MoE module implements the key MoE logics such as the router design, the balancing losses, or the expert-router interactions. Second, the training module optimization processes for all three training stages, which supports loading state-of-the-art LLMs, handling custom datasets, hyper-parameter configurations, the sparse upcyclying algorithms, and the

main training loop. Lastly, the evaluation module is our custom implementation of LMMS-Eval to support almost 100 benchmarks and various metrics proposed in this work. Figure 1 depicts the design of LibMoE, showing its key modules and their interactions.

Overall, LibMoE offers a modular, highly extendable implementation to study MoE. LibMoE empowers users to tailor and experiment with different MoE configurations, supporting rapid prototyping and fair evaluation of novel methodologies without the necessity for extensive reconfiguration of the underlying framework. More importantly, by incorporating MoE to state-of-the-art LLMs, we ensure that future algorithms based on LibMoE will have high utility in real-world, large scale scenarios.

#### 4 EXPERIMENTS

Stage	Tokens					
Suge	Image	Text	Total			
Pre-Training	3.21e8	1.52e7	3.37e8			
Pre-FineTuning	4.08e8	1.59e8	5.67e8			
VIT (332K)	1.80e8	7.71e7	2.57e8			
VIT (665K)	3.60e8	1.54e8	5.14e8			

*Table 1.* Token distribution across different stages. VIT denotes Visual Instruction Tuning, with 332K and 665K indicating the number of images used.

#### 4.1 Experiment Settings

**Training task and datasets** We consider the vision-language pretraining task (Lu et al., 2019) and follow CUMO (Li et al., 2024b) to upcycle the LLava model (Liu et al., 2023a), allowing us to evaluate various MoE algo-

LIBMoE: A Library for comprehensive benchmarking Mixture of Experts in Large Language Models

Data	Model	MoE Method	AI2D	Text VQA	GQA	Hallusion Benchmark	MathVista Validation	MMBenchEN dev	MMMU Validation	MMStar	POPE	SQA Full	MME	AVEGAGE (w/o MME)
	CLIP + Phi3	SMoE-R	63.67	47.47	59.46	43.32	31.60	66.67	40.11	37.94	86.87	77.23	1,608.21	55.42
		Cosine-R	63.31	48.83	59.25	41.54	31.80	67.96	39.56	39.09	86.81	76.96	1,637.99	55.51
		Sigmoid-R	63.80	47.74	59.24	41.43	31.40	68.30	40.78	38.70	87.49	77.61	1,611.36	55.65
		Hyper-R	64.05	47.76	59.61	41.11	32.50	69.24	41.33	39.27	86.68	77.31	1,602.59	55.89
		Perturbed Cosine-R	64.60	47.92	59.08	41.54	30.60	67.87	40.22	38.84	86.81	77.82	1,619.69	55.63
	SigLIP 224 +	SMoE-R	65.19	39.39	59.55	40.69	29.80	68.99	40.00	40.88	85.88	79.08	1,688.78	54.94
		Cosine-R	65.12	40.78	59.41	40.48	31.50	70.10	40.00	40.84	86.58	79.21	1,719.35	55.40
332k	Phi3	Sigmoid-R	64.48	40.29	59.10	40.06	30.50	69.67	40.89	39.97	86.39	78.81	1,684.78	55.02
332K	Pm3	Hyper-R	65.15	40.57	58.82	40.80	30.50	70.62	40.56	40.59	85.82	81.66	1,692.64	55.51
		Perturbed Cosine-R	65.09	41.09	59.61	40.48	31.60	70.02	40.78	40.72	85.86	79.67	1,707.34	55.49
		SMoE-R	64.93	39.33	59.16	41.22	30.60	70.19	41.89	41.67	85.22	79.56	1,659.30	55.38
	SigLIP 224 + Phi3.5	Cosine-R	65.54	41.11	60.14	42.17	31.30	71.65	42.00	41.83	85.58	79.79	1,709.32	56.11
		Sigmoid-R	63.99	39.17	59.29	42.48	31.50	70.10	41.67	42.27	85.68	79.71	1,693.87	55.58
		Hyper-R	64.67	40.71	59.69	42.59	30.60	70.70	42.44	41.47	85.74	79.63	1,697.50	55.83
		Perturbed Cosine-R	64.96	40.63	59.76	42.17	32.00	71.05	41.89	41.72	86.03	79.77	1,711.27	56.00
	GI ID	SMoE-R	64.25	46.57	62.12	40.48	31.00	68.12	39.89	37.13	87.50	77.74	1,700.61	55.48
		Cosine-R	64.51	49.79	61.38	40.80	31.30	67.01	40.67	39.36	87.52	77.48	1,687.37	55.98
	CLIP +	Sigmoid-R	64.38	47.12	61.65	40.80	31.90	67.87	40.11	39.20	86.93	77.17	1,710.42	55.71
	Phi3	Hyper-R	64.37	47.59	59.70	40.38	31.30	68.30	40.78	38.33	85.70	80.33	1,726.87	55.68
		Perturbed Cosine-R	64.70	47.16	61.90	39.43	32.80	69.50	39.89	40.33	87.42	77.64	1,672.70	56.08
	SigLIP 224 + Phi3	SMoE-R	64.35	40.35	60.03	41.75	28.70	67.96	40.22	39.47	84.31	80.71	1,655.81	54.78
		Cosine-R	64.60	41.98	60.74	41.43	31.30	70.61	41.22	38.50	86.33	81.49	1,759.21	55.82
665k		Sigmoid-R	64.66	41.05	60.52	40.80	28.80	69.07	40.89	39.29	86.54	80.85	1,766.03	55.25
OUSK		Hyper-R	65.12	41.67	59.88	41.32	30.30	69.33	41.44	39.86	85.40	79.03	1,752.39	55.34
		Perturbed Cosine-R	64.80	41.89	61.00	40.90	31.80	70.70	42.00	39.60	86.43	81.44	1,776.54	56.06
		SMoE-R	65.71	41.21	60.78	42.38	29.70	70.79	42.11	41.67	85.19	79.13	1,665.15	55.87
	SigLIP 224 + Phi3.5	Cosine-R	65.16	41.92	61.15	41.96	29.20	70.79	42.78	41.57	86.39	79.42	1,748.67	56.03
		Sigmoid-R	64.60	40.49	60.23	41.01	30.40	70.45	41.44	42.05	86.03	78.54	1,684.87	55.52
		Hyper-R	65.06	41.68	60.97	41.01	31.40	70.88	42.22	42.60	86.78	79.43	1,707.86	56.20
		Perturbed Cosine-R	65.54	41.85	61.04	41.75	30.50	71.65	43.00	41.72	86.73	78.88	1,688.82	56.27

Table 2. Comparison of MoE algorithms on different models and training data sizes for visual instruction tuning. The data set is constructed from LLaVA-665K (Liu et al., 2023a). We highlight the highest (best) results in bold. Model: We consider five algorithms: SMoE-R (SMoE Router) (Shazeer et al., 2017), Cosine-R (Chi et al., 2022), Sigmoid-R (Sigmoid Router) (Csordás et al., 2023), Hyper-R (Hyper Router) (Do et al., 2023), and Perturbed Cosine-R (Perturbed Cosine Router) (Nguyen et al., 2024a)

rithms. In the dense training stage, we use the LLaVA-558K dataset (Liu et al., 2023a) to initialize the MLP connector and then train all three components using the ALLaVA dataset (Chen et al., 2024b). After this stage, we obtain a dense checkpoint that is ready to to upcycle to become various MoE models. For the MoE training stage, we utilize the full LLaVA-665K dataset (Liu et al., 2023a). Additionally, we also subsample the LLaVA-665K dataset to only keep half of the training samples (332K). The 332K dataset allows us to quickly gauge the algorithms behaviors and also supports fast debugging for future development. On the other hand, training on the full 665K dataset will report the algorithm scalability when more training data available. Table 4 reports the number of tokens available in each stage. For the purpose of benchmarking MoE algorithms, we adopt the standard datasets and configurations across all experiments. For a detailed analysis of the data sources and the purpose of each training stage, we refer to the original work of Liu et al. (2023a); Li et al. (2024b).

Evaluation Benchmarks and Metrics LibMoE employs a set of popular benchmarks for vision-language models, including AI2D (Kembhavi et al., 2016), ChartQA (Masry et al., 2022), Text VQA (Singh et al., 2019), GQA (Hudson, 2019), Hallusion Benchmark (Guan et al., 2023), Math-Vista (Lu et al., 2023), MMBenchEN (Liu et al., 2023b),

MME (Fu et al., 2023), MMMU (Yue et al., 2023), MMStar (Chen et al., 2024c), POPE (Li et al., 2023), and Science-QA (Lu et al., 2022). We carefully choose these benchmark to assert the model across several vision-language capabilities such as perception, reasoning, OCR, instruction following, and more. Beyond the performance on standard benchmarks, we analyze the algorithms holistically by evaluating their generalization throughout training, expert selection behaviors, and expert specialization, which we will report in Section 4.2.2.

**MoE algorithm** We implement five state-of-the-art MoE algorithms for comparison, First, we consider the dense MoE with a sigmoid router (Csordás et al., 2023), which has been shown to achieve a better convergence rate than the standard softmax. Then, we include the original SMoE (Fedus et al., 2022) that uses a softmax router and a load balancing loss. Then, we implement three SMoE algorithms such as the cosine router (Chi et al., 2022) and its pertubed variant (Nguyen et al., 2024a), and lastly the hyper-router (Do et al., 2023). These algorithms covers both the dense and sparse variants of MoE, have strong theoretical guarantees, and have shown encouraging results on classical benchmarks. For the sigmoid router, we always active two experts, while only two out four experts are activated per sample in the sparse algorithms (K = 2 and N = 4).

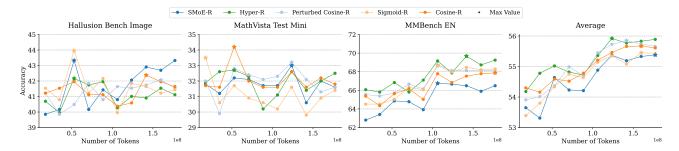


Figure 3. Comparison of the performance of different MoE algorithms over time. The experiments are conducted on the **LLaVa-332K** dataset and the CLIP + Phi3 model.



Figure 4. Impact of Training Data Percentage on Expert Selection.

Model architecture We consider Phi3 mini (Abdin et al., 2024) and Phi3.5 mini (Abdin et al., 2024) for the LLM to reduce the training time while achieving good performances. For the visual encoder, we use either CLIP (Radford et al., 2021) or Siglip (Zhai et al., 2023), which are two common choices. We follow Li et al. (2024b) to set the learning rate schedule, checkpointing frequency across all experiment for a fair comparison. All the experiments presented here are readily supported by LibMoE.

#### 4.2 Main Results

#### 4.2.1 Performance Comparison

Table 2 reports the performances of the MoE algorithms considered, implemented by our LibMoE. We observe that there is not a clear winner consistently across all benchmarks. Although Hyper Router and Perturbed Cosine Router may perform slightly better in some settings, their improvements over the second best method is quite marginal. We argue that such a gap can happen because of the randomness in initialization, data shuffling, and will be diminished when increasing the amount of training data. Overall, using LibMoE to compare state-of-the-art MoE algorithms under a fair and large scale setting, our result agrees with the common practice that the original SMoE strategy is still the most attractive choice thanks to its simplicity and scalability.

Generalization throughout training Next, we investigate the algorithms performances throughout training to get a more fine-grained understanding of their behaviors. To this end, we report the performance of the intermediate checkpoints (after each 10% of training data) in Figure 3. Due to space constrains, we only plot three representative benchmarks and the averaged of all 10 benchmarks in Figure 3 and provide the plot for all benchmarks in Figure 9, Appendix A. Interestingly, we observe that the last checkpoint is often not the one with the best performance. This result shows that although early stopping is not extensively studied in the literature, it can be beneficial for vision-language pre-training and could be a promising research direction.

#### 4.2.2 Expert Selection Analysis

Beyond the empirical performance, we conduct extensive analyses to understand the considered algorithms' behaviors. Particularly, we are interested in how the experts are selected throughout training according the different algorithms. Our findings reveal that the five MoE algorithms demonstrate distinct characteristics in adapting to these conditions, offering a novel perspective on their dynamic behavior. In the following, we will investigate the expert selection patterns according to various scenarios.

# (a) Analyzing the Impact of Training Data Proportions on Expert Selection in MoE Algorithms

In this experiment, we examine how varying training data

proportions influence expert selection dynamics within specific layers of several Sparse Mixture-of-Experts (SMoE) algorithms. As illustrated in Figure 4, the rate of change in expert selection gradually decreases as the dataset size increases, with discrepancies in expert allocation between consecutive 10% data segments narrowing progressively. Particularly in models trained on 90% versus 100% of the dataset, the difference in expert allocation drops to approximately 0-13%, suggesting that MoE algorithms effectively stabilize expert allocation disparities as data volume grows. This observation implies that with larger datasets, MoE algorithms enhance their efficiency in assigning appropriate experts to distinct input patterns.

When focusing on the 90% and 100% data segments where both the Cosine Router and Perturbed Cosine Router display near-zero changes in expert selection rates, indicating a high degree of stability and consistency in expert routing across the MMBench EN, MMStar, and ScienceQA IMG benchmarks. The Perturbed Cosine Router, in particular, exhibits minimal changes in early training stages, followed by stronger fluctuations around the 70-80% training range, before stabilizing close to 0% by the end. This pattern indicates that the Perturbed Cosine Router achieves an earlier convergence in its routing strategy compared to other MoE algorithms, adapting its expert selection more efficiently as training progresses. Supporting these findings, the performance metrics summarized in Table 2 show that the Perturbed Cosine Router consistently outperforms other MoE algorithms when trained on the LLAVA-665K dataset. This combination of robust early performance and rapid stabilization suggests that the Perturbed Cosine Router is highly effective for tasks requiring stable, efficient expert utilization.

Figure 4 further elucidates the differing rates of expert selection changes across equivalent dataset sizes among various MoE algorithms, including the Perturbed Cosine Router, Cosine Router, Hyper Router, and SMoE Router. Although all MoE algorithms exhibit fluctuations in expert selection, there is a clear trend toward reduced variation over time, suggesting a process of **structural convergence** within these architectures. This structural convergence reflects an increasingly specialized alignment of each expert with specific subtasks, becoming particularly evident in the later training stages (e.g., beyond 80% of the dataset), where expert selection rates stabilize across layers. This stabilization indicates that the MoE algorithms effectively optimize their expert utilization, resulting in consistent routing patterns and enhanced specialization across the network.

### (b) Expert Responsiveness for Capacity Specialization on Subtasks

We investigate the behaviors of the expert selection mechanism by exploring how often each expert is selected in the

MME benchmark.

To this end, we analyze the frequency of the expert selection across different subtasks to gain insights into the specialization behavior of each expert. Given an MoE algorithm with N experts and L layers, the selection frequency of each expert i at a given layer l is denoted as  $\operatorname{freq}_i^{(l)}$  ( $i=1,2,\ldots,N$  and  $l=1,2,\ldots,L$ ). Note that this selection frequency is counted across all samples in the benchmark, in this case we choose to be MME. Then, the entropy  $H^{(l)}$  at each layer l is calculated by integrating the probability of selecting expert i into Shannon's entropy formula as follows:

$$H^{(l)} = -\sum_{i=1}^{N} \left( \frac{\operatorname{freq}_{i}^{(l)}}{\sum_{j=1}^{N} \operatorname{freq}_{j}^{(l)}} \right) \log_{2} \left( \frac{\operatorname{freq}_{i}^{(l)}}{\sum_{j=1}^{N} \operatorname{freq}_{j}^{(l)}} \right), \tag{4}$$

where:

- $\text{freq}_i^{(l)}$ : The number of times expert i is selected at layer l.
- N: The total number of experts in the MoE algorithms.
- $\sum_{j=1}^{N} \operatorname{freq}_{j}^{(l)}$ : The total number of expert selections at layer l.
- $H^{(l)}$ : The entropy value at layer l, measuring the uncertainty or diversity in expert selections.

For a single metric to evaluate the diversity in expert utilization throughout the entire model, we propose  $H_{\rm total}$  - the total entropy which averages of the entropy values across the L layers as:

$$H_{\text{total}} = \frac{1}{L} \sum_{l=1}^{L} H^{(l)}.$$
 (5)

With  $H_{\rm total}$ , we can measure the frequency of expert selections across all layers  $H_{\rm total}$ . Importantly, high  $H_{\rm total}$  Indicates a balanced expert utilization across all layers, where the model tends to distribute selections evenly among all experts. In contrast, low  $H_{\rm total}$  Suggests a concentrated usage of a few experts in most layers, indicating specialization or a preference for certain experts.

Figure 5 report the overall entropy  $H_{\text{total}}$  of each algorithm, organized into different subtasks. we observe a common trend that complex subtasks such as text\_translation, code\_reasoning, and numerical\_calculation tend to yield relatively low entropy scores compared to other subtasks. Notably, the SMoE Router and Hyper Router exhibit similar

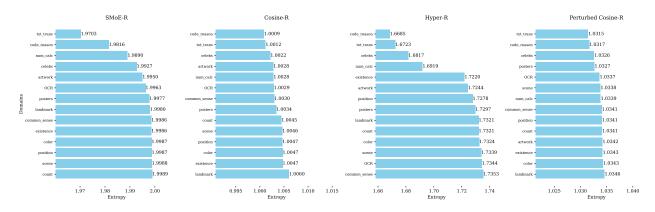


Figure 5. Entropy analysis of expert selection frequency across subtasks in MoE algorithms. The entropy values indicate the tendency of different routers to consistently select specific experts for given subtasks.

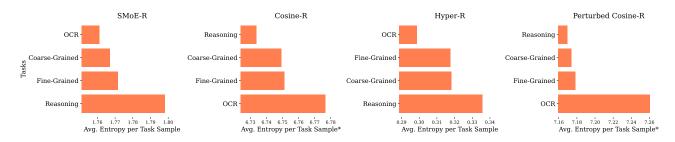


Figure 6. Measured confidence levels of various MoE algorithms across tasks. Entropy was computed for each sample and then averaged within each task to illustrate differences in confidence across MoE algorithms. For the Cosine-R and Perturbed Cosine-R algorithms, values on the x-axis (denoted by \*) were scaled to enhance visualization of subtle entropy variations. The scaled entropy values are calculated using the transformation (entropy -1.999)  $\times 10000$ .

entropy distributions for these specialized tasks, with the difference between their minimum and maximum entropy scores being 0.03 and 0.07, respectively. This range is substantially larger than that observed in the Cosine Router and Perturbed Cosine Router, suggesting that SMoE and Hyper Routers tend to specialize by engaging only a select subset of experts for these more complex tasks. Conversely, for less complex subtasks, entropy scores are higher, indicating that processing is distributed across a wider range of experts, likely enhancing generalization across various subtasks.

In contrast, the Cosine Router and Perturbed Cosine Router demonstrate smaller entropy variations, with differences between their minimum and maximum entropy scores limited to just 0.0051 and 0.0031, respectively. This reduced entropy range suggests a more consistent specialization, with specific experts engaged regardless of the subtask type. Such behavior implies that the Cosine Router and Perturbed Cosine Router maintain a higher specialization capacity, effectively activating designated experts even across varied tasks, which may benefit tasks requiring a higher degree of precision and domain-specific expertise.

Additionally, the Cosine Router and Perturbed Cosine

Router consistently maintain low entropy values across all subtasks, often close to 1. This pattern indicates a high degree of specialization, with experts focusing intently on specific subtasks. Such low entropy suggests that these routers have learned to isolate knowledge, concentrating it within designated domains and thus minimizing knowledge diffusion across unrelated tasks. Notably, while both routers exhibit similarly low entropy, they diverge in their areas of maximum specialization: the Cosine Router demonstrates its lowest entropy in code reasoning, whereas the Perturbed Router is most specialized in text translation. These variations demonstrate distinct specialization behaviors among the considered MoE algorithms.

### (c) Does Overconfidence in Gated Expert Routing Impact MoE Performance?

In Figure 6, we assess the confidence levels of various MoE algorithms by examining the entropy in expert probability distributions across samples. This entropy is averaged per task (OCR Task, Coarse-Grained Tasks, and Reasoning Tasks) to reveal trends in confidence within each MoE configuration. The results display distinct behavioral patterns among routers, especially in their probability distributions

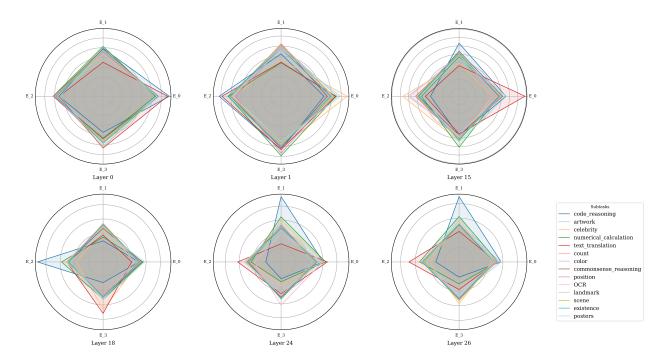


Figure 7. Expert selection across layers on different tasks in the MME benchmarks. The model uses SigLIP as the vision encoder and Phi 3.5 as the LLM. This figure highlights the distinct expert selection behavior observed in the vision encoder layers.

#### across experts.

The SMoE Router and Hyper Router demonstrate lower entropy for perception-oriented tasks, such as the OCR Task, implying strong confidence in expert selection for these data types. For more cognitively challenging tasks, such as Reasoning, both routers show a significant increase in entropy, indicating a broader distribution across multiple experts. This variability in entropy between perception and reasoning tasks suggests that the SMoE Router and Hyper Router exhibit higher confidence in simpler, perceptual tasks, possibly due to the straightforward nature of the input. Notably, the Hyper Router maintains the lowest entropy overall, a result of its phased training strategy that begins with top-1 expert selection and later transitions to top-2. This phased approach appears to bolster the Hyper Router's confidence in expert assignments.

In contrast, the Cosine Router and Perturbed Cosine Router exhibit a different trend, displaying their lowest entropy in the Reasoning Task, with entropy values that are consistently higher than those of the SMoE Router and Hyper Router. This pattern suggests a more uniform allocation of probabilities across experts, thus avoiding strong biases toward any single expert. The minimal variation in entropy values across tasks for these routers (within  $1e^{-5}$ ) indicates a consistently balanced distribution of probabilities among experts, with negligible differences across tasks. This bal-

anced behavior in the Cosine Router and Perturbed Cosine Router likely mitigates the risk of overcommitting to specific experts, thereby maintaining flexibility and adaptability across diverse tasks.

Interestingly, despite the higher confidence (indicated by low entropy) exhibited by the SMoE Router and Hyper Router, their overall performance can sometimes be worse than that of the Cosine and Perturbed Routers, particularly when trained with more data (Table 2, LLAVA-665K dataset). This discrepancy may be attributed to an overconfidence effect, whereby the SMoE Router and Hyper Router concentrate probabilities heavily on a primary expert, resulting in the underutilization of auxiliary experts. Such an overreliance on a single expert diminishes the advantages of using multiple experts, failing to harness the potential diversity and complementary strengths available within an ensemble. Our experiments illustrate that excessive confidence in expert selection can inadvertently reduce the effectiveness of using many experts, highlighting the importance of balanced probability distributions to fully leverage the potential of MoE architectures.

#### (d) Transitioning from Generalization to Specialization: Layer-Wise Expert Selection

A key question we investigate is the behavior of expert selection across different layers. To this end, we follow Xue et al. (2024) to report the frequency of expert selection across

different layers. In Figure 7, we plot the expert selection frequencies at different layers in the MME benchmark, where we choose two shallow, two middle, and two deep layers in the visual encoder. Our findings reveal that expert assignments fluctuate more prominently in the middle layers, with increased stability and convergence in the later layers. The early layers (e.g., layers 0 and 1) display initial uniformity, where the model processes subtasks in a generalized manner with limited specialization. However, as we move toward the middle layers, particularly around layers 15 and 16, a clearer distinction between subtasks emerges, accompanied by more fluctuations in expert selection at this stage. This marks a critical transition, where experts are chosen more selectively based on the specific characteristics of each subtask. In the final layers, such as layers 23 and 24, the gating mechanism becomes more stable and specialized for each subtask, with no significant differences observed in the frequency of expert selection across subtasks. This trend reveals the model's gradual shift from generalization to a more refined, subtask-specific specialization as it progresses through the network.

### 4.2.3 Impact of the architectural choice on expert selection

Lastly, we examine the impact of the architecture choices to the expert selection pattern.

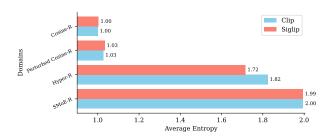


Figure 8. Comparison of the average entropy of the frequency distribution of selected experts across subtasks using different vision encoders: Siglip and CLIP.

As illustrated in Figure 8, the Cosine Router and Perturbed Cosine Router display minimal differences in average entropy between the CLIP and SigLIP encoders. However, with the SMoE Router and Hyper Router, the differences become more pronounced. SigLIP exhibits slightly lower average entropy scores than CLIP, suggesting an enhanced ability to extract nuanced visual features. This capability enables SigLIP to dynamically allocate experts more effectively, adapting to the unique complexity of each subtask and handling diverse input distributions more robustly.

Accordingly, as shown in Table 2, models utilizing SigLIP achieve nearly equivalent performance to CLIP-based mod-

els while operating at a lower resolution (224 vs. 336) and requiring less training time. Detailed training times of the model configurations are provided in Appendix B. These findings highlight the importance of the vision encoder in revealing the behaviors of MoE algorithms in expert routing and in promoting expert specialization across various subtasks.

#### 4.3 Summary of Key Results

Using LibMoE, we have extensively evaluate five state-ofthe-art MoE algorithm. The empirical results show that no algorithms perform consistently the best when averaging across 11 zero-shot benchmarks. However, a closer look at their training curves reveals that there exists an intermediate checkpoint that can achieve much better performances than the commonly used last checkpoint. However, we choose to not use the best checkpoint as the algorithm's performance because of the lack of a reliable early-stopping mechanism and the expensive cost of evaluating all intermediate checkpoints across many benchmarks. Then, we investigate the expert selection behaviors and show that different algorithms often have distinct patterns when faced with tasks of different characteristics such as perception or reasoning tasks. Furthermore, we show that over-confident in expert selection might not always be optimal. Lastly, we highlight the impact of the architecture choice where Siglip is a better backbone for SMoE than CLIP. Our empirical results not only agree with the common practice for training large scale sMoE models but also reveal promising research direction for future studies. Lastly, we emphasize that all experiments conducted are readily supported by our LibMoE and we will also provide all the dense training checkpoints to facilitate the future research.

#### 5 CONCLUSION

In this work, we present LiBMoE, a specialized toolkit developed to advance research in Mixture-of-Experts (MoE) algorithms. LibMoE offers a streamlined approach to training and evaluation MoE algorithms with LLMs, while being efficient and accessible to many researchers. LibMoE offers a suite of algorithms, standardized training settings, and a wide-range of benchmarks, all of which can be easily customized and extendable by the community. We benchmark five state-of-the-art MoE algorithms extensively and provide a comprehensive study of their unique characteristics, many of which are not discussed in the literature. We firmly believe that LibMoE will serve as an indispensable tool for researchers to study MoE.

#### REFERENCES

- Abdin, M., Jacobs, S. A., Awan, A. A., Aneja, J., Awadallah, A., Awadalla, H. H., Bach, N., Bahree, A., Bakhtiari, A., Behl, H. S., Benhaim, A., Bilenko, M., Bjorck, J., Bubeck, S., Cai, M., Mendes, C. C. T., Chen, W., Chaudhary, V., Chopra, P., Giorno, A. D., de Rosa, G., Dixon, M., Eldan, R., Iter, D., Goswami, A., Gunasekar, S., Haider, E., Hao, J., Hewett, R. J., Huynh, J., Javaheripi, M., Jin, X., Kauffmann, P., Karampatziakis, N., Kim, D., Kim, Y. J., Khademi, M., Kurilenko, L., Lee, J. R., Lee, Y. T., Li, Y., Liang, C., Liu, W., Lin, E., Lin, Z., Madan, P., Mitra, A., Modi, H., Nguyen, A., Norick, B., Patra, B., Perez-Becker, D., Portet, T., Pryzant, R., Qin, H., Radmilac, M., Rosset, C., Roy, S., Saarikivi, O., Saied, A., Salim, A., Santacroce, M., Shah, S., Shang, N., Sharma, H., Song, X., Ruwase, O., Vaddamanu, P., Wang, X., Ward, R., Wang, G., Witte, P., Wyatt, M., Xu, C., Xu, J., Yadav, S., Yang, F., Yang, Z., Yu, D., Zhang, C.-Y., Zhang, C., Zhang, J., Zhang, L. L., Zhang, Y., Zhang, Y., and Zhou, X. Phi-3 technical report: A highly capable language model locally on your phone. ArXiv, abs/2404.14219, 2024. URL https://api.semanticscholar. org/CorpusID:269293048.
- Chen, G., Zhao, X., Chen, T., and Cheng, Y. Moerbench: Towards building reliable language models with sparse mixture-of-experts. *ArXiv*, abs/2406.11353, 2024a. URL https://api.semanticscholar.org/CorpusID:270560405.
- Chen, G. H., Chen, S., Zhang, R., Chen, J., Wu, X., Zhang, Z., Chen, Z., Li, J., Wan, X., and Wang, B. Allava: Harnessing gpt4v-synthesized data for a lite vision-language model. arXiv preprint arXiv:2402.11684, 2024b.
- Chen, L., Li, J., wen Dong, X., Zhang, P., Zang, Y., Chen, Z., Duan, H., Wang, J., Qiao, Y., Lin, D., and Zhao, F. Are we on the right way for evaluating large vision-language models? *ArXiv*, abs/2403.20330, 2024c. URL https://api.semanticscholar.org/CorpusID:268793433.
- Chen, S., Jie, Z., and Ma, L. Llava-mole: Sparse mixture of lora experts for mitigating data conflicts in instruction finetuning mllms. *arXiv preprint arXiv:2401.16160*, 2024d.
- Chi, Z., Dong, L., Huang, S., Dai, D., Ma, S., Patra, B., Singhal, S., Bajaj, P., Song, X., Mao, X.-L., et al. On the representation collapse of sparse mixture of experts. *Advances in Neural Information Processing Systems*, 35: 34600–34613, 2022.
- Csordás, R., Irie, K., and Schmidhuber, J. Approximating two-layer feedforward networks for efficient transformers. *arXiv* preprint arXiv:2310.10837, 2023.

- Dai, D., Dong, L., Ma, S., Zheng, B., Sui, Z., Chang, B., and Wei, F. Stablemoe: Stable routing strategy for mixture of experts. *arXiv preprint arXiv:2204.08396*, 2022.
- Dai, D., Deng, C., Zhao, C., Xu, R., Gao, H., Chen, D., Li, J., Zeng, W., Yu, X., Wu, Y., et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*, 2024.
- Do, G., Le, K., Pham, Q., Nguyen, T., Doan, T.-N., Nguyen, B. T., Liu, C., Ramasamy, S., Li, X., and Hoi, S. Hyperrouter: Towards efficient training and inference of sparse mixture of experts. *arXiv preprint arXiv:2312.07035*, 2023.
- Du, N., Huang, Y., Dai, A. M., Tong, S., Lepikhin, D., Xu, Y., Krikun, M., Zhou, Y., Yu, A. W., Firat, O., et al. Glam: Efficient scaling of language models with mixture-ofexperts. In *International Conference on Machine Learn*ing, pp. 5547–5569. PMLR, 2022.
- Fedus, W., Zoph, B., and Shazeer, N. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.
- Fu, C., Chen, P., Shen, Y., Qin, Y., Zhang, M., Lin, X., Qiu, Z., Lin, W., Yang, J., Zheng, X., Li, K., Sun, X., and Ji, R. Mme: A comprehensive evaluation benchmark for multimodal large language models. *ArXiv*, abs/2306.13394, 2023. URL https://api.semanticscholar.org/CorpusID:259243928.
- Guan, T., Liu, F., Wu, X., Xian, R., Li, Z., Liu, X., Wang, X., Chen, L., Huang, F., Yacoob, Y., Manocha, D., and Zhou, T. Hallusionbench: An advanced diagnostic suite for entangled language hallucination and visual illusion in large vision-language models. 2023. URL https://api.semanticscholar.org/CorpusID:265499116.
- Han, X., Nguyen, H., Harris, C., Ho, N., and Saria, S. Fusemoe: Mixture-of-experts transformers for fleximodal fusion. *arXiv preprint arXiv:2402.03226*, 2024.
- He, J., Qiu, J., Zeng, A., Yang, Z., Zhai, J., and Tang, J. Fastmoe: A fast mixture-of-expert training system. *arXiv* preprint arXiv:2103.13262, 2021.
- Huang, Q., An, Z., Nan, Z., Tao, M., Zhang, C., Jin, Y., Xu, K., Chen, L., Huang, S., and Feng, Y. Harder tasks need more experts: Dynamic routing in moe models. ArXiv, abs/2403.07652, 2024. URL https://api.semanticscholar. org/CorpusID:268363693.

- Hudson, D. A. Gqa: A new dataset for real-world visual reasoning and compositional question answering - supplementary material. 2019. URL https: //api.semanticscholar.org/CorpusID: 268114221.
- Hwang, C., Cui, W., Xiong, Y., Yang, Z., Liu, Z., Hu, H., Wang, Z., Salas, R., Jose, J., Ram, P., et al. Tutel: Adaptive mixture-of-experts at scale. *Proceedings of Machine Learning and Systems*, 5:269–287, 2023.
- Jacobs, R. A., Jordan, M. I., Nowlan, S. J., and Hinton, G. E. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.
- Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., de Las Casas, D., Hanna, E. B., Bressand, F., Lengyel, G., Bour, G., Lample, G., Lavaud, L. R., Saulnier, L., Lachaux, M.-A., Stock, P., Subramanian, S., Yang, S., Antoniak, S., Scao, T. L., Gervet, T., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. Mixtral of experts. ArXiv, abs/2401.04088, 2024. URL https://api.semanticscholar.org/CorpusID:266844877.
- Kembhavi, A., Salvato, M., Kolve, E., Seo, M., Hajishirzi, H., and Farhadi, A. A diagram is worth a dozen images. *ArXiv*, abs/1603.07396, 2016. URL https://api.semanticscholar.org/CorpusID:2682274.
- Komatsuzaki, A., Puigcerver, J., Lee-Thorp, J., Ruiz, C. R., Mustafa, B., Ainslie, J., Tay, Y., Dehghani, M., and Houlsby, N. Sparse upcycling: Training mixtureof-experts from dense checkpoints. arXiv preprint arXiv:2212.05055, 2022.
- Lepikhin, D., Lee, H., Xu, Y., Chen, D., Firat, O., Huang, Y., Krikun, M., Shazeer, N., and Chen, Z. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv* preprint arXiv:2006.16668, 2020.
- Li, D., Liu, Y., Wu, H., Wang, Y., Shen, Z., Qu, B., Niu, X., Wang, G., Chen, B., and Li, J. Aria: An open multimodal native mixture-of-experts model. 2024a. URL https://api.semanticscholar.org/CorpusID:273229053.
- Li, J., Wang, X., Zhu, S., Kuo, C.-W., Xu, L., Chen, F., Jain, J., Shi, H., and Wen, L. Cumo: Scaling multimodal llm with co-upcycled mixture-of-experts. *arXiv preprint arXiv:2405.05949*, 2024b.
- Li, Y., Du, Y., Zhou, K., Wang, J., Zhao, W. X., and rong Wen, J. Evaluating object hallucination in large vision-language models. In *Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://api.semanticscholar.org/CorpusID:258740697.

- Lian, X., Jacobs, S. A., Kurilenko, L., Tanaka, M., Bekman, S., Ruwase, O., and Zhang, M. Universal checkpointing: Efficient and flexible checkpointing for large scale distributed training. *ArXiv*, abs/2406.18820, 2024. URL https://api.semanticscholar.org/CorpusID:270764954.
- Lin, X. V., Shrivastava, A., Luo, L., Iyer, S., Lewis, M., Gosh, G., Zettlemoyer, L. S., and Aghajanyan, A. Moma: Efficient early-fusion pre-training with mixture of modality-aware experts. *ArXiv*, abs/2407.21770, 2024. URL https://api.semanticscholar.org/CorpusID:271571529.
- Liu, H., Li, C., Wu, Q., and Lee, Y. J. Visual instruction tuning. In *NeurIPS*, 2023a.
- Liu, Y., Duan, H., Zhang, Y., Li, B., Zhang, S., Zhao, W., Yuan, Y., Wang, J., He, C., Liu, Z., Chen, K., and Lin, D. Mmbench: Is your multi-modal model an all-around player? *ArXiv*, abs/2307.06281, 2023b. URL https://api.semanticscholar.org/CorpusID:259837088.
- Lu, J., Batra, D., Parikh, D., and Lee, S. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Neural Information Processing Systems*, 2019. URL https://api.semanticscholar.org/CorpusID: 199453025.
- Lu, P., Mishra, S., Xia, T., Qiu, L., Chang, K.-W., Zhu, S.-C., Tafjord, O., Clark, P., and Kalyan, A. Learn to explain: Multimodal reasoning via thought chains for science question answering. *ArXiv*, abs/2209.09513, 2022. URL https://api.semanticscholar.org/CorpusID:252383606.
- Lu, P., Bansal, H., Xia, T., Liu, J., yue Li, C., Hajishirzi, H., Cheng, H., Chang, K.-W., Galley, M., and Gao, J. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. In *In*ternational Conference on Learning Representations, 2023. URL https://api.semanticscholar. org/CorpusID:264491155.
- Masry, A., Long, D. X., Tan, J. Q., Joty, S. R., and Hoque, E. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. *ArXiv*, abs/2203.10244, 2022. URL https://api.semanticscholar.org/CorpusID: 247593713.
- Muennighoff, N., Soldaini, L., Groeneveld, D., Lo, K., Morrison, J. D., Min, S., Shi, W., Walsh, P., Tafjord, O., Lambert, N., Gu, Y., Arora, S., Bhagia, A., Schwenk, D., Wadden, D., Wettig, A., Hui, B., Dettmers, T.,

- Kiela, D., Farhadi, A., Smith, N. A., Koh, P. W., Singh, A., and Hajishirzi, H. Olmoe: Open mixture-of-experts language models. *ArXiv*, abs/2409.02060, 2024. URL https://api.semanticscholar.org/CorpusID:272366674.
- Nguyen, H., Akbarian, P., Pham, T., Nguyen, T., Zhang, S., and Ho, N. Statistical advantages of perturbing cosine router in sparse mixture of experts. *arXiv preprint arXiv:2405.14131*, 2024a.
- Nguyen, H., Ho, N., and Rinaldo, A. Sigmoid gating is more sample efficient than softmax gating in mixture of experts. *ArXiv*, abs/2405.13997, 2024b. URL https://api.semanticscholar.org/CorpusID:269983353.
- Nguyen, H., Ho, N., and Rinaldo, A. On least squares estimation in softmax gating mixture of experts. *arXiv* preprint arXiv:2402.02952, 2024c.
- Pham, Q., Do, G., Nguyen, H., Nguyen, T., Liu, C., Sartipi, M., Nguyen, B., Ramasamy, S., Li, X., Hoi, S. C. H., and Ho, N. Competesmoe effective training of sparse mixture of experts via competition. *ArXiv*, abs/2402.02526, 2024. URL https://api.semanticscholar.org/CorpusID:267411820.
- Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.
- Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv* preprint arXiv:1701.06538, 2017.
- Singh, A., Natarajan, V., Shah, M., Jiang, Y., Chen, X., Batra, D., Parikh, D., and Rohrbach, M. Towards vqa models that can read. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8309–8318, 2019. URL https://api.semanticscholar.org/CorpusID:85553602.
- Sukhbaatar, S., Golovneva, O., Sharma, V., Xu, H., Lin, X. V., Rozière, B., Kahn, J., Li, S.-W., tau Yih, W., Weston, J., and Li, X. Branch-train-mix: Mixing expert llms into a mixture-of-experts llm. *ArXiv*, abs/2403.07816, 2024. URL https://api.semanticscholar.org/CorpusID:268363969.
- Wei, T., Zhu, B., Zhao, L., Cheng, C., Li, B., Lü, W., Cheng, P., Zhang, J., Zhang, X., Zeng, L., et al. Skywork-moe: A deep dive into training techniques for mixture-of-experts language models. arXiv preprint arXiv:2406.06563, 2024.

- Xie, Y., Huang, S., Chen, T., and Wei, F. Moec: Mixture of expert clusters. In AAAI Conference on Artificial Intelligence, 2022. URL https://api.semanticscholar.org/CorpusID: 250644033.
- Xue, F., Zheng, Z., Fu, Y., Ni, J., Zheng, Z., Zhou, W., and You, Y. Openmoe: An early effort on open mixture-of-experts language models. *arXiv preprint arXiv:2402.01739*, 2024.
- Yang, A., Yang, B., Hui, B., Zheng, B., Yu, B., Zhou, C., Li, C., Li, C., Liu, D., Huang, F., Dong, G., Wei, H., Lin, H., Tang, J., Wang, J., Yang, J., Tu, J., Zhang, J., Ma, J., Xu, J., Zhou, J., Bai, J., He, J., Lin, J., Dang, K., Lu, K., Chen, K.-Y., Yang, K., Li, M., Xue, M., Ni, N., Zhang, P., Wang, P., Peng, R., Men, R., Gao, R., Lin, R., Wang, S., Bai, S., Tan, S., Zhu, T., Li, T., Liu, T., Ge, W., Deng, X., Zhou, X., Ren, X., Zhang, X., Wei, X., Ren, X., Fan, Y., Yao, Y., Zhang, Y., Wan, Y., Chu, Y., Cui, Z., Zhang, Z., and Fan, Z.-W. Qwen2 technical report. ArXiv, abs/2407.10671, 2024. URL https://api.semanticscholar.org/CorpusID:271212307.
- Yue, X., Ni, Y., Zhang, K., Zheng, T., Liu, R., Zhang, G., Stevens, S., Jiang, D., Ren, W., Sun, Y., Wei, C., Yu, B., Yuan, R., Sun, R., Yin, M., Zheng, B., Yang, Z., Liu, Y., Huang, W., Sun, H., Su, Y., and Chen, W. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. *ArXiv*, abs/2311.16502, 2023. URL https://api.semanticscholar.org/CorpusID:265466525.
- Zhai, X., Mustafa, B., Kolesnikov, A., and Beyer, L. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11975–11986, 2023.
- Zhang, K., Li, B., Zhang, P., Pu, F., Cahyono, J. A., Hu, K., Liu, S., Zhang, Y., Yang, J., Li, C., and Liu, Z. Lmms-eval: Reality check on the evaluation of large multimodal models, 2024. URL https://arxiv.org/abs/2407.12772.
- Zhou, Y., Lei, T., Liu, H., Du, N., Huang, Y., Zhao, V., Dai, A. M., Le, Q. V., Laudon, J., et al. Mixture-ofexperts with expert choice routing. *Advances in Neural Information Processing Systems*, 35:7103–7114, 2022.

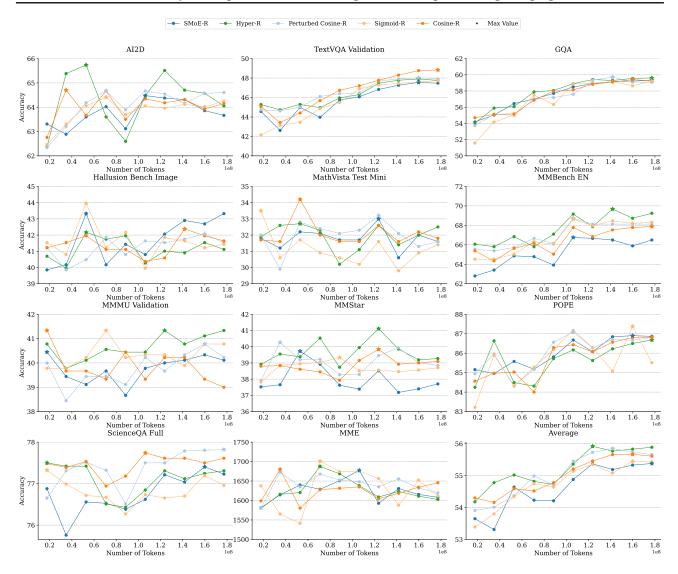


Figure 9. Comparison of the performance of different MoE algorithms across 11 benchmarks over time. The experiments were conducted using the LLaVa-332K dataset and the CLIP + Phi3 model.

## A THE PERFORMANCE OF DIFFERENT MOE ALGORITHMS OVER TIME.

In addition to the results presented in Figure 3, Figure 9 offers a detailed view of the time-dependent performance of five MoE algorithms across 11 benchmarks. This figure illustrates the unique behavioral characteristics of each algorithm and supports our observation that, in most cases, the final checkpoints of the MoE algorithms do not necessarily yield the best performance. This finding underscores the potential benefits of applying early stopping to achieve optimal results.

#### **B** TRAINING TIMES DETAILS

Table 3 summarizes training times for various Mixture-of-Experts (MoE) algorithms within the LibMoE frame-work, across different models and datasets. It includes pre-training, fine-tuning, and visual instruction tuning times for each method on the CLIP + Phi3, Siglip 224 + Phi3, and Siglip 224 + Phi3.5 models. Algorithms such as SMoE-R, Cosine-R, Sigmoid-R, Hyper-R, and Perturbed Cosine-R were trained with different GPU configurations (from 4 - 6 A100 GPUs) on datasets of varying sizes (332K to 665K samples).

Model	Metho	d	Time training	Resource	Dataset
	Pre-Trair	ning	5h 35m	4xA100	558K
	Pre-FineTu	ıning	18h	4xA100	708K
		SMoE-R	50h	6xA100	665K
	Visual Instruction Tuning	Cosine-R	55h	6xA100	665K
		Sigmoid-R	54h	6xA100	665K
		Hyper-R	48h	6xA100	665K
CLIP + Phi3		Perturbed Cosine-R	55h	6xA100	665K
		SMoE-R	32h	4xA100	332K
	Visual Instruction Tuning	Cosine-R	33h	4xA100	332K
		Sigmoid-R	30h	4xA100	332K
		Hyper-R	27h	4xA100	332K
		Perturbed Cosine-R	34h	4xA100	332K
	Pre-Train	ning	2h 45m	4xA100	558K
	Pre-FineTu	ıning	16h	4xA100	708K
	Visual Instruction Tuning	SMoE-R	44h	6xA100	665K
		Cosine-R	45h	6xA100	665K
		Sigmoid-R	42h	6xA100	665K
		Hyper-R	41h	6xA100	665K
Siglip 224 + Phi3		Perturbed Cosine-R	45h	6xA100	665K
	Visual Instruction Tuning	SMoE-R	31h	4xA100	332K
		Cosine-R	32h	4xA100	332K
		Sigmoid-R	30h	4xA100	332K
		Hyper-R	26h	4xA100	332K
		Perturbed Cosine-R	31h	4xA100	332K
	Pre-Train	ning	2h 55m	4xA100	558K
	Pre-FineTu	ıning	17h 05m	4xA100	708K
		SMoE-R	43h	6xA100	665K
	Visual Instruction Tuning	Cosine-R	44h	6xA100	665K
		Sigmoid-R	43h	6xA100	665K
		Hyper-R	36h	6xA100	665K
<b>Siglip 224 + Phi3.5</b>		Perturbed Cosine-R	44h	6xA100	665K
		SMoE-R	32h	4xA100	332K
		Cosine-R	33h	4xA100	332K
	Visual Instruction Tuning	Sigmoid-R	32h	4xA100	332K
		Hyper-R	25h	4xA100	332K
		Perturbed Cosine-R	32h	4xA100	332K

Table 3. Detailed Training Duration and Resource Utilization for MoE Algorithms Across Models and Datasets