LLaMA-MoE: Building Mixture-of-Experts from LLaMA with Continual Pre-Training

Tong Zhu[1*] **, Xiaoye Qu**[2]**, Daize Dong**[2]**, Jiacheng Ruan**[3]**, Jingqi Tong**[4]**,**
Conghui He[2]**, Yu Cheng**[5]

[1] Soochow University [2] Shanghai AI Laboratory [3] Shanghai Jiao Tong University
[4] Fudan University [5] The Chinese University of Hong Kong

tzhu7@stu.suda.edu.cn, {quxiaoye,dongdaize.d,heconghui}@pjlab.org.cn,
jackchenruan@sjtu.edu.cn, jqtong23@m.fudan.edu.cn, chengyu@cse.cuhk.edu.hk

Abstract

Mixture-of-Experts (MoE) has gained increasing popularity as a promising framework for scaling up large language models (LLMs). However, training MoE from scratch in a large-scale setting still suffers from data-hungry and instability problems. Motivated by this limit, we investigate building MoE models from existing dense large language models. Specifically, based on the well-known LLaMA-2 7B model, we obtain an MoE model by: (1) *Expert Construction*, which partitions the parameters of original Feed-Forward Networks (FFNs) into multiple experts; (2) *Continual pre-training*, which further trains the transformed MoE model and additional gate networks. In this paper, we comprehensively explore different methods for expert construction and various data sampling strategies for continual pre-training. After these stages, our LLaMA-MoE models could maintain language abilities and route the input tokens to specific experts with part of the parameters activated. Empirically, by training 200B tokens, LLaMA-MoE-3.5B models significantly outperform dense models that contain similar activation parameters.

1 Introduction

Large language models (LLMs) (ChatGPT, 2023; Touvron et al., 2023; Su et al., 2024b,a; Lu et al., 2024b,a) have presented remarkable understanding and reasoning capability on a wide range of tasks. Nowadays, scaling model size has become the de facto approach to augment performance efficacy further. However, the immense model size is unsustainable due to the computational costs. Inspired by this, we focus on sparsely activated Mixture-of-Expert (MoE) models that decouple model size from computation costs.

Training MoE from scratch (Lepikhin et al., 2020; Fedus et al., 2022; Zoph et al., 2022; Xue et al., 2024; Dai et al., 2024) leads to a significant overall budget. In this work, we reduce the training costs by investigating building MoE models from existing dense LLMs. Moreover, starting from the dense model provides flexible structure design choices for MoE. In other words, we can place MoE in any transformer block. In this paper, we are dedicated to building a full MoE model, where each layer contains an MoE block.

To build strong LLaMA-MoE models, we identify two important challenges. First, how to effectively construct experts from the Feed-Forward Networks (FFNs) in the existing LLMs. There are works exploring splitting FFN parameters to construct experts (Zhang et al., 2021; Zuo et al., 2022) on T5 or BERT model. Conversely, Komatsuzaki et al. (2022) directly copy the FFNs to form experts. However, there is no existing work exploring it for decoder-only models. Notably, the FFN structure of the previous T5 or BERT model is based on the ReLU function, which shares significantly different characteristics from recent LLMs based on SwiGLU function. Second, overcoming the performance decrease entailed by changing the network structure from dense to sparse remains challenging. Due to the reduction in the amount of activated parameters and the newly introduced gate network for expert routing, we observe a significant performance drop between the LLaMA-MoE models and the original dense LLaMA models.

To solve the above issues, we comprehensively explore four different methods for expert construction. Among them, the non-overlapping randomly splitting method achieves the best performance. Subsequently, we continue training the transformed MoE models and additional gate networks. In this stage, we also carefully study both dynamic and static data sampling strategies for obtaining the fastest convergence and performance improvement. Finally, with a static domain weight proportion corresponding to the activated parameters, the

[*]Work was done during an internship at Shanghai AI Laboratory. Code and models are available at https://github.com/pjlab-sys4nlp/llama-moe

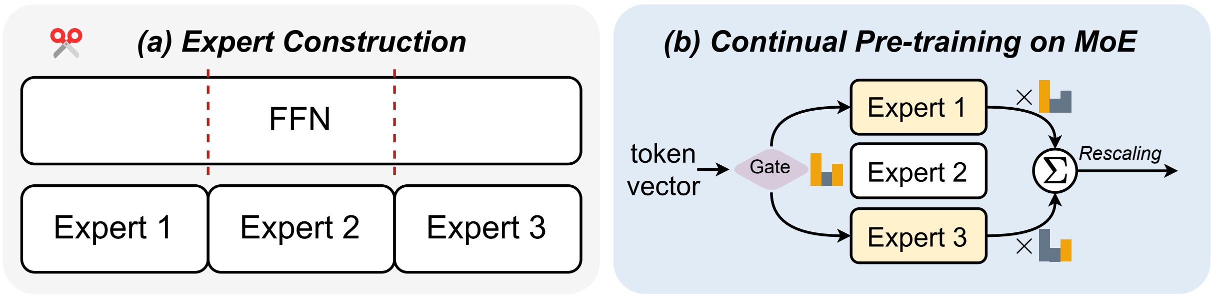


Figure 1: The main framework of building LLaMA-MoE models. (a) The original FFNs in the LLaMA are split into different experts. (b) In the transformed LLaMA-MoE, the hidden states are processed by partially chosen experts instead of all experts. We continue to train the LLaMA-MoE to improve the performance.

LLaMA-MoE models can quickly converge to a decent level with 200B tokens.

In summary, our contributions are as follows:

- We propose a framework to develop mixture-of-experts from existing decoder-style LLMs by splitting FFNs and continual pretraining, which has never been explored before.
- We comprehensively explore different splitting methods for expert construction. Meanwhile, we comprehensively investigate both dynamic and static data sampling strategy for continual pretraining.
- Our extensive experiments on a variety of tasks validate the effectiveness of our proposed LLaMA-MoE series models. Notably, all our model construction processes and training data are transparent.

2 Method

As illustrated in Figure 1, we construct LLaMA-MoE from LLaMA-2-7B by first partitioning FFNs into multiple experts and each token is routed to top-k experts. Continual pre-training is subsequently applied to recover the MoE model's language ability. The following sections describe the details of our method.

2.1 Expert Construction

Splitting FFN. We start with the feed-forward network (FFN) in LLaMA which uses SwiGLU (Shazeer, 2020) as the activation function. Each FFN layer in LLaMA consists of three parts: an up projection weight $W_{\text{up}} \in \mathbb{R}^{d\times d_h}$, a gate projection weight $W_{\text{gate}} \in \mathbb{R}^{d\times d_h}$ and a down projection weight $W_{\text{down}} \in \mathbb{R}^{d_h\times d}$. Given the universal set U containing indices of all intermediate neurons $\{1, 2, \ldots, d_h\}$, based on whether the indices are shared among different experts, we implement two groups of construction methods: Neuron-Independent and Neuron-Sharing. Specifically, we devise four methods to construct experts: (1) **Independent**$_{\text{Random}}$ randomly divides neurons into non-overlapping groups; (2) **Independent**$_{\text{Clustering}}$ groups neurons according to clustering results; (3) **Sharing**$_{\text{Inner}}$ assigns neurons to experts based on pre-clustered data importance vectors; (4) **Sharing**$_{\text{Inter}}$ creates shared neurons as independent blocks while distributing others via importance. More details are presented in Appendix C. In this paper, we adopt the **Independent**$_{\text{Random}}$ which uniformly splits U into non-overlapping indices sets $S_1, S_2, \ldots, S_n$ and construct n experts with each size $m = \frac{d_h}{n}$. After this stage, we can build LLaMA-MoE models with n experts.

Rescaling. After partitioning a dense FFN layer into multiple small experts, the activated expert parameters are much smaller than the original dense models. To preserve the representational capacity of the partitioned model, we introduce a scale factor and apply rescale operations to guarantee effective expert output. In particular, considering activating k out of n experts, we scale the output of expert by a factor of $\frac{n}{k}$.

2.2 Continual Pre-training

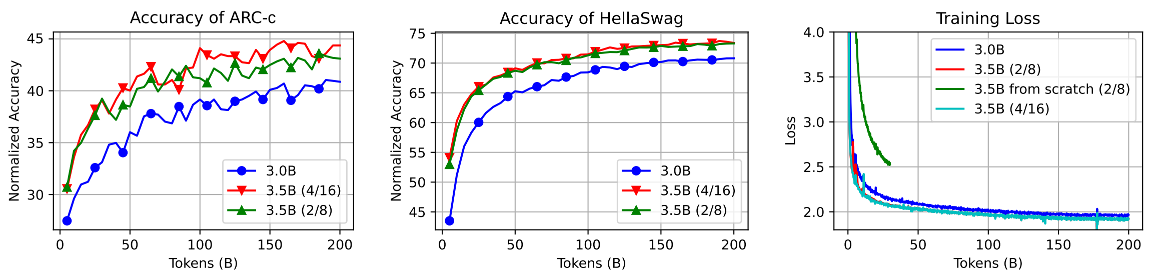
Since the original LLaMA model structure is reorganized to MoE, we continue pre-training the LLaMA-MoE model to recover its language ability. The training objective is the same as LLaMA-2 (Touvron et al., 2023).

Data Sampling Strategies. The data sampling weights are crucial to obtain a global optimum (Xie et al., 2023). Thus, we investigate both static and dynamic data sampling strategies including (1) **Static**$_{\text{Sheared}}$ fixes the sampling weights to Sheared-

Model	Commonsense & Reading Comprehension					
	SciQ	PIQA	WinoGrande	ARC-e	ARC-c (25)	HellaSwag (10)
LLaMA-2-7B	93.7	78.1	69.3	76.4	53.0	78.6
OPT-2.7B	78.9	74.8	60.8	54.4	34.0	61.4
Pythia-2.8B	83.2	73.6	59.6	58.8	36.7	60.7
INCITE-Base-3B	85.6	73.9	63.5	61.7	40.3	64.7
Open-LLaMA-3B-v2	88.0	**77.9**	63.1	63.3	40.1	71.4
Sheared-LLaMA-2.7B	87.5	76.9	65.0	63.3	41.6	71.0
LLaMA-MoE-3.0B (2/16)	84.2	77.5	63.6	60.2	40.9	70.8
LLaMA-MoE-3.5B (4/16)	87.6	**77.9**	65.5	**65.6**	**44.2**	**73.3**
LLaMA-MoE-3.5B (2/8)	**88.4**	77.6	**66.7**	65.3	43.1	**73.3**

Model	Continued		LM	World Knowledge		Average
	LogiQA	BoolQ (32)	LAMBADA	NQ (32)	MMLU (5)	
LLaMA-2-7B	30.7	82.1	73.9	28.0	46.6	64.6
OPT-2.7B	25.8	63.3	63.6	10.7	25.8	50.3
Pythia-2.8B	28.1	65.9	64.6	8.7	26.8	51.5
INCITE-Base-3B	27.5	65.8	65.4	15.2	27.2	53.7
Open-LLaMA-3B-v2	28.1	69.2	67.4	16.0	26.8	55.6
Sheared-LLaMA-2.7B	28.3	73.6	68.3	17.6	**27.3**	56.4
LLaMA-MoE-3.0B (2/16)	**30.6**	71.9	66.6	17.0	26.8	55.5
LLaMA-MoE-3.5B (4/16)	29.7	**75.0**	**69.5**	**20.3**	26.8	**57.7**
LLaMA-MoE-3.5B (2/8)	29.6	73.9	69.4	19.8	27.0	57.6

Table 1: Main results on downstream tasks. LLaMA-MoE-3.0B (2/16) means the activated parameters are 3.0B and 2 out of 16 experts are activated. The shot number used is noted in parentheses, with 0-shot if not specified.



(a) ARC-c (25) (b) HellaSwag (10) (c) Training loss

Figure 2: Model performances on ARC-c and HellaSwag dataset and the training loss for LLaMA-MoE-3.0B and LLaMA-MoE-3.5B. The two models are trained with 200B tokens.

LLaMA (Xia et al., 2023) throughout the training process; (2) **Static**$_{\text{LLaMA}}$ utilizes the static sampling weights of LLaMA (Touvron et al., 2023); (3) **Dynamic**$_{\text{Sheared}}$ follows (Xia et al., 2023) to initialize all the sampling weights to the same ones and updates every 5B tokens with comparing the loss differences between LLaMA-MoE and LLaMA-2-7B; (4) **Dynamic**$_{\text{LLaMA}}$ is similar to **Dynamic**$_{\text{Sheared}}$ but initialized with the LLaMA sampling weights. In this paper, we use **Static**$_{\text{Sheared}}$ as the data sampling strategy.

3 Experiments

3.1 Datasets and Baselines

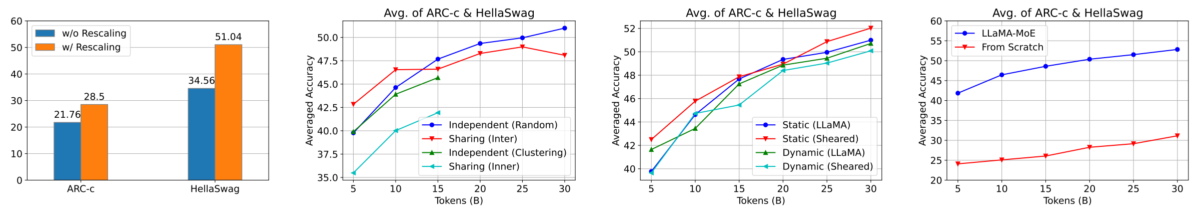
The continual pretraining dataset for LLaMA-MoE is SlimPajama (Soboleva et al., 2023), which contains 627B tokens from seven domains. More implementation details are in Appendix D. For comprehensive ability assessment, we follow Xia et al. (2023). The detailed evaluation dataset can be found in Appendix E. We compare LLaMA-MoE with strong pre-trained language models containing similar activation parameters, including OpenLLaMA-3B-v2 (Geng and Liu, 2023), OPT-2.7B (Zhang et al., 2022), Pythia-2.8B (Biderman et al., 2023), INCITE-Base-3B (TogetherAI, 2023), and Sheared LLaMA (Xia et al., 2023).

3.2 Main Results

As shown in Table 1, LLaMA-MoE-3.5B (2/8) and LLaMA-MoE-3.5B (4/16) achieve similar average results and the latter is slightly better. However,

Model	Open LLM Leaderboard					Alignment
	MMLU	ARC-c	HellaSwag	TruthfulQA	Avg.	MT-Bench
Sheared-LLaMA-2.7B-ShareGPT	**28.41**	41.04	71.21	47.65	47.08	3.79
Sheared-LLaMA-2.7B (Our Dataset)	25.24	43.69	71.70	**49.00**	47.41	4.06
LLaMA-MoE-v1-3.0B (2/16)	23.61	43.43	72.28	44.24	45.89	4.15
LLaMA-MoE-v1-3.5B (4/16)	26.49	**48.29**	**75.10**	45.91	**48.95**	4.60
LLaMA-MoE-v1-3.5B (2/8)	25.53	45.99	74.95	44.39	47.71	**4.72**

Table 2: Supervised fine-tuned model performances on Open LLM Leaderboard tasks and open-ended questions. Sheared LLaMA-2.7B-ShareGPT is a chat model created by Xia et al. (2023). We reimplement the chat model by instruction tuning on our dataset and provide fair comparisons.



(a) Rescaling (b) Expert Construction (c) Data Sampling Weights (d) MoE from Scratch

Figure 3: Four ablation studies exploring the most important components in our LLaMA-MoE framework. Limited by the training budget, in all figures, we stop training the specific model variants when an obvious trend emerges.

LLaMA-MoE-3.5B significantly surpasses open-source models with similar activation parameters. Specifically, LLaMA-MoE-3.5B (4/16) exceeds the competitive model Sheared LLaMA by 1.3 average points. Meanwhile, LLaMA-MoE-3.0B performs comparably with Open-LLaMA-3B-v2. We also find LLaMA-MoE-3.5B (4/16) could achieve 89.2% of the average performance compared with the original LLaMA-2-7B, validating the effectiveness.

To demonstrate the training progress and model capability changes, in Figure 2 (a) and (b), we present the model performances on both ARC-c and HellaSwag and find the results grow gradually as the training process goes on. For the training loss, as shown in Figure 2 (c), LLaMA-MoE-3.0B and LLaMA-MoE-3.5B converge to about 1.95 and 1.90, respectively. The final loss is higher than LLaMA-2 7B as these two models activate fewer parameters. Moreover, LLaMA-MoE converges much faster than training from scratch.

3.3 Ablation Study

In this section, we investigate four important components in our framework. As shown in Figure 3, (a) By training 5B tokens for model variants, we found that equipping with scale factor provides significantly better initial performance for MoE models. (b) Among four expert construction methods, after training 30B tokens, randomly splitting neurons into non-overlapping groups obtains the best performance. (c) Comparing different data sampling strategies, using the static sampling weights of Sheared-LLaMA achieves the best results. Although dynamic sampling shows performance improvements in Sheared-LLaMA, we find it hard to work for our models. (d) Our model significantly surpasses variants training from scratch, demonstrating the effectiveness of our framework for reducing the training budget.

3.4 Instruction Tuning

To evaluate the instructed MoE models' performances, we fine-tune LLaMA-MoE with curated 6k ShareGPT instruction data (Liu et al., 2023) for 2 epochs. As shown in Table 2, the instructed LLaMA-MoE-3.5B (4/16) outperforms the dense model on ARC-c (48.29 vs. 43.69) and HellaSwag (75.10 vs. 71.70) tasks. The overall performance on Open LLM Leaderboard [1] tasks surpasses the dense model (48.95 vs. 47.41). Besides, there is a large gap in alignment abilities, where LLaMA-MoE-3.5B (2/8) significantly outperforms Sheared LLaMA-2.7B by 0.66 scores on MT-Bench.

4 Conclusion

In this paper, we build MoE from a dense model by partitioning the FFN layers into experts, and

[1] https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard

continual pre-training. We comprehensively investigate different methods for expert construction and data sampling. Empirically, LLaMA-MoE-3.5B significantly outperforms open-source models with similar activation parameters. Meanwhile, LLaMA-MoE-3.0B achieves similar performance with Open-LLaMA-3B with less activated parameters. The instructed LLaMA-MoE models also present stronger abilities than their counterparts.

Limitations

Limited by the training budget, we construct MoE models on LLaMA2-7B model and continually pre-train them for 200B tokens. Although we have tested the method with three model settings (4/16E, 2/8E, and 2/16E), it is worth trying to investigate the scaling property with more experiments on the expert sizes, numbers, and training tokens. Moreover, due to the launch time of this project, we do not experiment on the latest open-source models, such as LLaMA3. In the future, we will apply our framework to more models.

Acknowledgments

We would like to show our sincere gratitude to all the reviewers for their insightful comments. We also appreciate Wenliang Chen, Peng Sun, Penglong Jiao, and Wenwen Qu for their supports and helpful discussions.

References

Mistral AI. 2023. Mixtral of experts: A high quality sparse mixture-of-experts.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

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

ChatGPT. 2023. Openai: Introducing chatgpt.

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

OpenCompass Contributors. 2023. Opencompass: A universal evaluation platform for foundation models. `https://github.com/open-compass/opencompass`.

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

Tri Dao. 2023. FlashAttention-2: Faster attention with better parallelism and work partitioning.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *The Journal of Machine Learning Research*, 23(1):5232–5270.

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. 2023. A framework for few-shot language model evaluation.

Xinyang Geng and Hao Liu. 2023. Openllama: An open reproduction of llama.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Aran Komatsuzaki, Joan Puigcerver, James Lee-Thorp, Carlos Riquelme Ruiz, Basil Mustafa, Joshua Ainslie, Yi Tay, Mostafa Dehghani, and Neil Houlsby. 2022. Sparse upcycling: Training mixture-of-experts from dense checkpoints. *arXiv preprint arXiv:2212.05055*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*.

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

Mike Lewis, Shruti Bhosale, Tim Dettmers, Naman Goyal, and Luke Zettlemoyer. 2021. Base layers:

Simplifying training of large, sparse models. In *International Conference on Machine Learning*, pages 6265–6274. PMLR.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *arXiv preprint arXiv:2007.08124*.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. 2023. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. *ArXiv*, abs/2312.15685.

Zhenyi Lu, Chenghao Fan, Wei Wei, Xiaoye Qu, Dangyang Chen, and Yu Cheng. 2024a. Twin-merging: Dynamic integration of modular expertise in model merging. *arXiv preprint arXiv:2406.15479*.

Zhenyi Lu, Jie Tian, Wei Wei, Xiaoye Qu, Yu Cheng, Dangyang Chen, et al. 2024b. Mitigating boundary ambiguity and inherent bias for text classification in the era of large language models. *arXiv preprint arXiv:2406.07001*.

Mikko I. Malinen and Pasi Fränti. 2014. Balanced k-means for clustering. In *Structural, Syntactic, and Statistical Pattern Recognition*, pages 32–41, Berlin, Heidelberg. Springer Berlin Heidelberg.

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

Samyam Rajbhandari, Conglong Li, Zhewei Yao, Minjia Zhang, Reza Yazdani Aminabadi, Ammar Ahmad Awan, Jeff Rasley, and Yuxiong He. 2022. Deepspeed-moe: Advancing mixture-of-experts inference and training to power next-generation ai scale. In *International Conference on Machine Learning*, pages 18332–18346. PMLR.

Stephen Roller, Sainbayar Sukhbaatar, Jason Weston, et al. 2021. Hash layers for large sparse models. *Advances in Neural Information Processing Systems*, 34:17555–17566.

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

Noam Shazeer. 2020. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*.

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

Daria Soboleva, Faisal Al-Khateeb, Robert Myers, Jacob R Steeves, Joel Hestness, and Nolan Dey. 2023. SlimPajama: A 627B token cleaned and deduplicated version of RedPajama.

Zhaochen Su, Juntao Li, Jun Zhang, Tong Zhu, Xiaoye Qu, Pan Zhou, Yan Bowen, Yu Cheng, et al. 2024a. Living in the moment: Can large language models grasp co-temporal reasoning? *arXiv preprint arXiv:2406.09072*.

Zhaochen Su, Jun Zhang, Tong Zhu, Xiaoye Qu, Juntao Li, Min Zhang, and Yu Cheng. 2024b. Timo: Towards better temporal reasoning for language models. *arXiv preprint arXiv:2406.14192*.

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

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

Johannes Welbl, Nelson F Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. *arXiv preprint arXiv:1707.06209*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. 2023. Sheared llama: Accelerating language model pre-training via structured pruning.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V. Le, Tengyu Ma, and Adams Wei Yu. 2023. Doremi: Optimizing data mixtures speeds up language model pretraining.

Fuzhao Xue, Zian Zheng, Yao Fu, Jinjie Ni, Zangwei Zheng, Wangchunshu Zhou, and Yang You. 2023. Openmoe: Open mixture-of-experts language models. https://github.com/XueFuzhao/OpenMoE.

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

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

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona T. Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models. *ArXiv*, abs/2205.01068.

Zhengyan Zhang, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021. Moefication: Transformer feed-forward layers are mixtures of experts. *arXiv preprint arXiv:2110.01786*.

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

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

Simiao Zuo, Qingru Zhang, Chen Liang, Pengcheng He, Tuo Zhao, and Weizhu Chen. 2022. Moebert: from bert to mixture-of-experts via importance-guided adaptation. *arXiv preprint arXiv:2204.07675*.

A Related Work

Mixture-of-Experts (MoE). Sparse models attempt to activate a subset of parameters for each input to save computation. In modern deep learning, the MoE architecture was first proven effective in LSTM (Shazeer et al., 2017), and later introduced to the transformer architecture as a substitute for the FFN layers (Lepikhin et al., 2020; Fedus et al., 2022). Subsequent studies explored the routing policies (Lewis et al., 2021; Roller et al., 2021; Zhou et al., 2022) and network architectures (Xue et al., 2023; AI, 2023) of MoE. Our work follows Shazeer et al. (2017) and implement the token-level noisy top-k gating with load balancing loss.

Expert Construction. There are two lines of works constructing MoE from dense checkpoints. The first category splits the parameters of the FFNs and ensures that the total model parameters remain unchanged (Zuo et al., 2022; Zhang et al., 2021). Another type of work expands the total model parameters while keeping the activation parameters as the original dense models (Komatsuzaki et al., 2022). Our work follows the first research line and decomposes the original FFNs into multiple small experts. Different from previous works, we focus on a SwiGLU-based decoder-style models and continues training the MoE models.

B Preliminary

A standard Mixture of Experts (MoE) layer comprises N expert networks $\{E_1, E_2, \ldots, E_N\}$ and a gating network G which activates the top-k experts and distributes input tokens to corresponding experts. Formally, given an input embedding x, the MoE layer's output is the sum of outputs from k selected experts:

$$y = \sum_{i \in \mathcal{K}} G(x)_i \cdot E_i(x), \tag{1}$$

where the indices set $\mathcal{K}$ are determined by $G(x)$, and $E_i(x)$ denotes the output of the i-th expert.

C Expert Construction

Based on whether the intermediate neurons within the FFN are shared among different experts, we implement two groups of construction methods: *Neuron-Independent* and *Neuron-Sharing*.

Neuron-Independent. We formulate expert construction as a task of partitioning into equal-sized sets. Given a universal set U containing indices of all intermediate neurons $\{1, 2, \ldots, d_h\}$, we uniformly split U into n equal-sized indices set $S_1, S_2, \ldots, S_n$ and construct experts with size $m = \frac{d_h}{n}$, where we have:

$$\bigcup_{i=1}^{n} S_i = U \quad \text{and} \quad \bigcap_{i=1}^{n} S_i = \varnothing. \tag{2}$$

Specifically, we describe two kinds of partition methods:

- **Independent**$_{\text{Random}}$: We randomly partition U into n equal-sized subsets.
- **Independent**$_{\text{Clustering}}$: Following (Zhang et al., 2021), we perform a balanced k-means clustering (Malinen and Fränti, 2014) with n centroids on the row vectors of W_{up} and partition U according to the clustering result.

Neuron-Sharing. According to (Zuo et al., 2022), the representation ability of a model can be partially retained through a structured partition. Therefore, we treat the expert construction as a structured pruning problem, by measuring the first-order Taylor expansion on loss change ΔL for each intermediate neuron when it gets pruned. For each FFN layer, we maintain a vector $v \in \mathbb{R}^{d_h}$ initialized as zeros to record the importance of its intermediate neurons. Given batched data D, the importance vector v is updated as follows:

$$v := v + \sum_{(x,y)\in D} \left| h \odot \nabla_h L(x, y) \right|. \quad (3)$$

The indices sets $S_1, S_2, \ldots, S_n$ are then generated using certain algorithm for the experts with sizes $m = \frac{d_h}{n}$. Given the universal indices set $U = \{1, 2, \ldots, d_h\}$, we have:

$$\bigcup_{i=1}^{n} S_i \in U. \quad (4)$$

- **Sharing**$_{\text{Inner}}$: We obtain n importance vectors $v_1, v_2, \ldots, v_n$ through pre-clustered n groups of data. For each expert i, the corresponding S_i consists the indices of neurons with the largest m values in v_i.

- **Sharing**$_{\text{Inter}}$: Referencing the implementation in (Rajbhandari et al., 2022), we set aside the neurons shared by most experts as independent residual blocks, while others are assigned according to the importance vectors $v_1, v_2, \ldots, v_n$.

D Implementation Details

All models are trained on 112 A100 (80G) GPUs with a global batch size of 15M tokens for 13.6K steps (total 200B tokens). The context length is 4096. The maximum learning rate is 2e-4 with 100 warmup steps and the final learning rate decays to 2e-5 with cosine scheduling. We construct three MoE models, 3B (2/16), 3.5B (4/16), and 3.5B (2/8) from LLaMA-2-7B. Here, 3B and 3.5B are the number of activated parameters, and "(2/16)" means 2 out of 16 experts are activated. Similar notations are applied to "(4/16)" and "(2/8)". Our implementation is based on transformers (Wolf et al., 2020), ZeRO-1 (Rajbhandari et al., 2022), and FlashAttention v2 (Dao, 2023).

E Evaluation datasets

We follow Xia et al. (2023) and use the lm-evaluation-harness (Gao et al., 2023) to evaluate the following downstream tasks: 0-shot normalized accuracy (acc_norm) of ARC Easy (Clark et al., 2018), LAMBADA (Paperno et al., 2016), LogiQA (Liu et al., 2020), PIQA (Bisk et al., 2020), SciQ (Welbl et al., 2017), and WinoGrande Standard (Sakaguchi et al., 2021), 10-shot HellaSwag (Zellers et al., 2019), 25-shot ARC Challenge (Clark et al., 2018), and 5-shot MMLU (Hendrycks et al., 2020). If there is no normalized accuracy, we use accuracy instead. Furthermore, we use OpenCompass (Contributors, 2023) to evaluate 32-shot NQ (Kwiatkowski et al., 2019).

F Expert Specialization

In this section, we present the expert specialization phenomenon we found in LLaMA-MoE. As Figure 4 shows, deep layers have more routing preferences than shallow layers. This may indicate that the shallow layers may capture more common features, while deep layers focus more on task-specific features. Based on this finding, expert partition on the latter layers' FFNs may bring further improvements. We leave it for future exploration. In deeper layers, each expert has different domain preferences and some experts are shared across different domains. These shared experts may represent data similarities among different domains. We also find the imbalance problem at the first two layers, where some experts are seldom selected. These experts may be pruned for future MoE model compression.

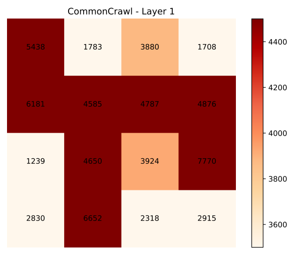
To investigate the latent correlations among domains, we normalize the number of routed tokens and calculate the L2 distances to represent the expert selection differences. As illustrated in Figure 5a, CommonCrwal and C4 datasets have similar expert preferences, while GitHub has similar expert preferences with arXiv and StackExchange. As to the Dev-to-Train differences in Figure 5b, we find HellaSwag and ARC-c share the most similar expert preferences with CommonCrawl and C4, and GSM-8K is similar to arXiv. This may provide some insights for continual pre-training to further improve downstream performances. For example, the model may consume more tokens from arXiv to improve GSM-8K results. However, expert selections on ARC-c and GSM-8K have greater distances with current pre-training data, which may involve new domains to deal with such tasks.

G Inference Efficiency

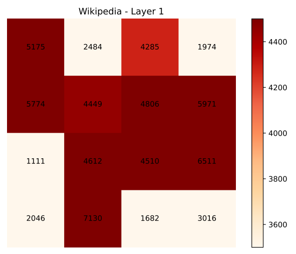
Table 3 demonstrates the inference computational cost of each model. We find the LLaMA-MoE-3.5B models consume only 57.7% FLOPs compared with LLaMA-2-7B, while the LLaMA-MoE-3.0B model only takes 50.7% FLOPs of LLaMA-2-7B, showing the inference efficiency.

Model	Inference TFLOPs
LLaMA-2-7B	62.9
LLaMA-MoE-3.0B (2/16)	31.9
LLaMA-MoE-3.5B (4/16)	36.3
LLaMA-MoE-3.5B (2/8)	36.3

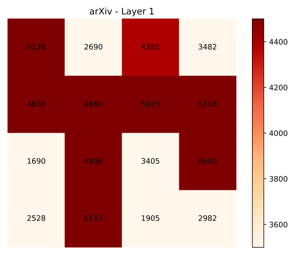
Table 3: Comparisons of model structure and inference efficiency. FLOPs are estimated with a sequence length of 4,096 and a batch size of 1.



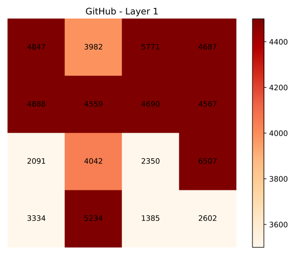
(a) CommonCrawl (1)



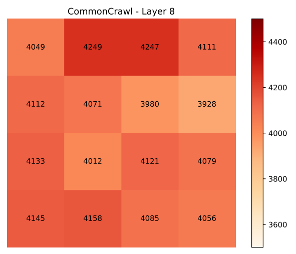
(b) Wikipedia (1)



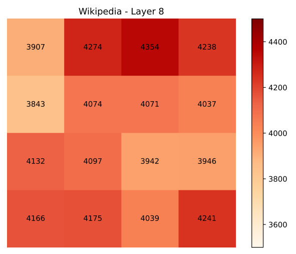
(c) arXiv (1)



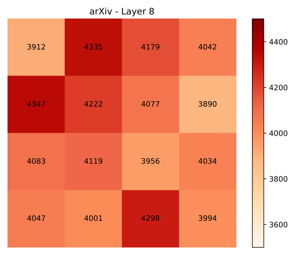
(d) GitHub (1)



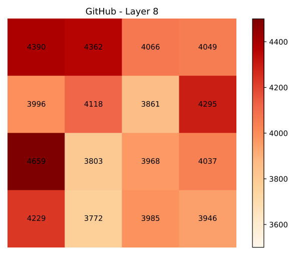
(e) CommonCrawl (8)



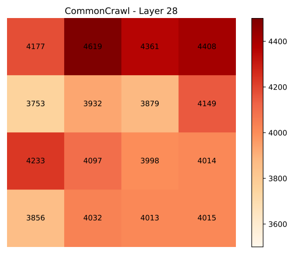
(f) Wikipedia (8)



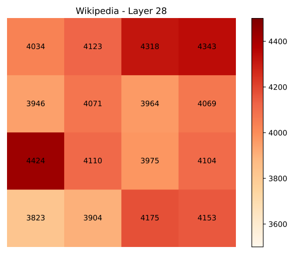
(g) arXiv (8)



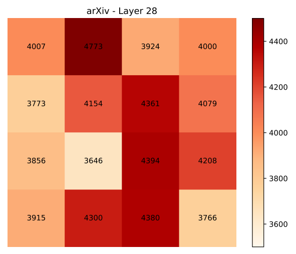
(h) GitHub (8)



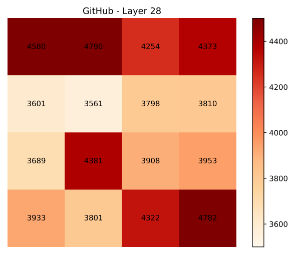
(i) CommonCrawl (28)



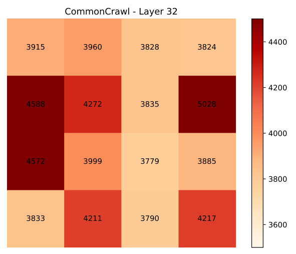
(j) Wikipedia (28)



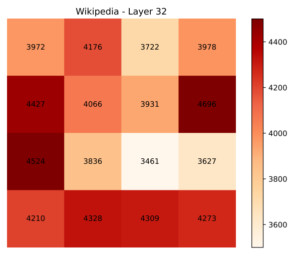
(k) arXiv (28)



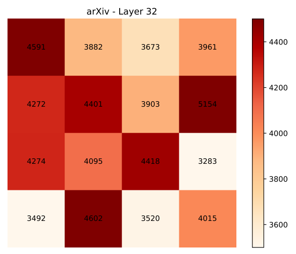
(l) GitHub (28)



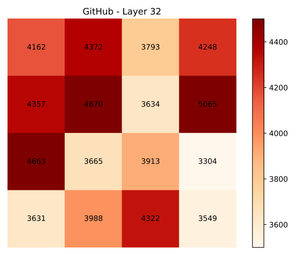
(m) CommonCrawl (32)



(n) Wikipedia (32)

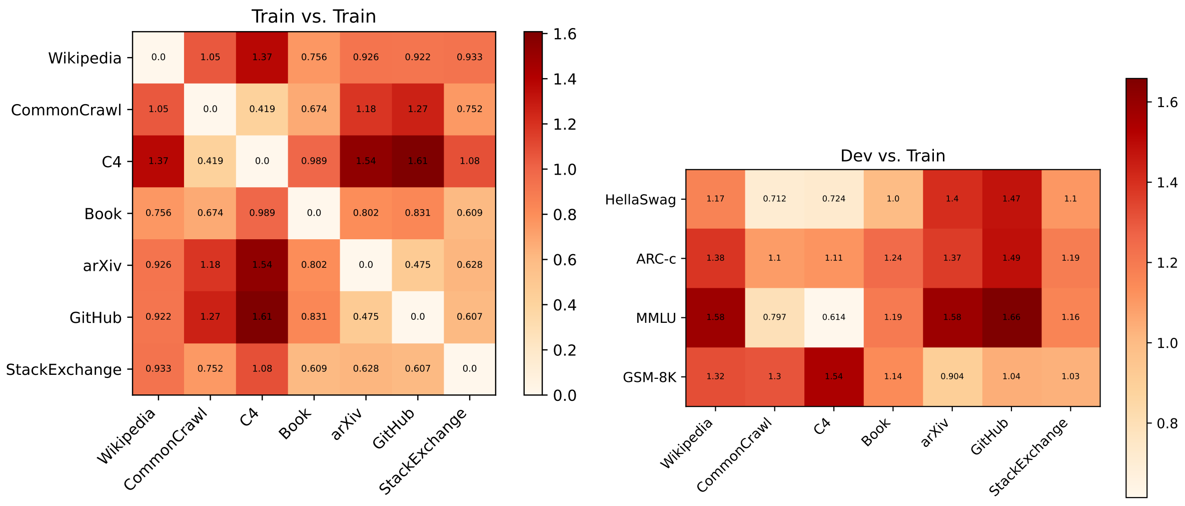


(o) arXiv (32)



(p) GitHub (32)

Figure 4: Expert routing statistics on the 1st, 8th, 28th, and 32nd layers for LLaMA-MoE-3.5B (4/16). Each cell represents the number of routed tokens to an expert. Our model has a total of 16 experts. We sample 65.5K tokens from each domain for this visualization.



(a) Train-Train

(b) Dev-Train

Figure 5: Expert routing differences at the 32nd layer. Smaller numbers and lighter colors represent more similar expert routing preferences. 8.4M tokens per domain are sampled for this experiment.