TOWARDS AN EMPIRICAL UNDERSTANDING OF MIXTURE OF EXPERTS DESIGN CHOICES

Dongyang Fan[⋆], **Bettina Messmer**[⋆], **Martin Jaggi**
EPFL, Switzerland
firstname.lastname@epfl.ch

ABSTRACT

In this study, we systematically evaluate the impact of common design choices in Mixture of Experts on validation performance, uncovering distinct influences when routing at a token or sequence level. We also present empirical evidence showing comparable performance between a learned router and a frozen, randomly initialized router, suggesting that learned routing may not be essential. Our study further reveals that Sequence-level routing can result in topic-specific weak expert specialization, in contrast to syntax specialization observed with Token-level routing. The topic understanding is independent of the languages used.

1 INTRODUCTION

The Mixture of Experts (MoEs) has been receiving unprecedented attention in the LLM era. While initially it has been proposed by Jacobs et al. (1991) to encourage expert specialization when the model is under-parameterized to fit the whole data domain, the contemporary practices (Fedus et al., 2022; Shazeer et al., 2017) do not specifically seek for expert specialization aspects, instead, they use MoE as a tool to scale up model expressiveness at a reduced inference cost. A study by Zoph et al. (2022a) revealed the existence of expert specialization in encoder blocks, particularly at a lexicon level. Furthermore, the recent Mistral paper by Jiang et al. (2024) provides evidence that the router exhibits structured syntactic behavior rather than topic-level understanding. We posit that the cultivation of fine-grained expert specialization is facilitated by Token-level routing mechanisms. At the sequence level, the router is compelled to consider contextual information. A natural question arises: *do experts acquire different knowledge bases when routing is performed at a sequence level?*

MoEs demonstrate exceptional performance, yet the specific design choices that contribute to this efficacy remain a subject of inquiry. In particular, the impact of these choices on Sequence-level routing's marginal validation performance is not well understood. Jiang et al. (2024) employs a Layer-wise Token-level Top-2 routing, a prevalent architectural choice in contemporary implementations. Meanwhile, Fedus et al. (2022) empirically substantiates the potential of Top-1 Token-level routing. We wish to *systematically ablate each of these design choices and discern the extent to which they contribute to model performance.*

In this study, we empirically investigate Mixture of Experts (MoE) training from scratch using GPT2 Small-scale models. While our findings may not generalize to larger Language Model scales due to computing constraints in academia, we aim to offer insights into MoEs. Our contributions include:

- Ablating common MoE design choices to quantify the marginal impact on validation performance.
- Demonstrating that design choice preferences might differ for Token- and Sequence-level routing.
- Justifying the existence of *weak expert* specialization in topics when routing at a sequence level.

2 RELATED WORKS

Design Choices Shazeer et al. (2017) proposed Noisy Top-K Gating and required $K > 1$ in order to receive gradients for the router network. Fedus et al. (2022) made Top-1 routing work by multiplying Top-1 gating probability with the corresponding gating output. Both works let the token

Codes available at https://github.com/epfml/towards_understanding_moe

Topk(K)	Design Choices		#Parameters	Validation Perplexity	
	Routing Unit	*Routing Strategy*	*(Total, Active)*	*MultiWiki*	*OpenWebText*
2	Token	Layer-wise	(295M, 182M)	**9.907**	**22.632**
	Sequence	Layer-wise		**10.667**	24.980
	Sequence	Global		11.605	25.689
1	Token	Layer-wise	(295M, 124M)	**10.188**	**23.548**
	Sequence	Layer-wise		11.573	26.742
	Sequence	Global		13.632	30.292
Baseline - $1\times$ FFN Width			(124M, 124M)	11.815	26.195
Baseline - $2\times$ FFN Width			(182M, 182M)	10.831	24.387
Baseline - $4\times$ FFN Width			(295M, 295M)	10.071	22.700

Table 1: Ablation studies of each component of MoE design choices. Number of experts (N) is set to 4 in all experiments. **Red** signifies surpassing the baseline model with the entire parameter count, while **green** indicates surpassing the baseline model with an equivalent active parameter count.

itself make a choice on which expert to go to, which usually leads to expert collapse. An auxiliary loss or special gating mechanism (Lewis et al., 2021) is needed to balance the load of the experts. While the most popular routing unit is token, some works explore sentence-level routing (Pham et al., 2023; Kudugunta et al., 2021) and task-level routing (Li et al., 2022; Fan et al., 2021). Task-level routing is predominantly used in multilingual translation settings, where an extra indicator for language pairs is used.

Expert Specialization Zoph et al. (2022a) found in an encoder-decoder network, only encoder experts exhibit specialization while decoder experts lack this ability. This specialization manifests primarily in syntactic features, encompassing punctuation, conjunctions, articles, and verbs. Jiang et al. (2024) corroborated the presence of structured syntactic behavior among experts but did not discern explicit patterns in expert assignments based on the topic. Xue et al. (2023) examined different levels of specialization and observed only context-independent specialization at a token ID level. Notably, the routing unit in these studies is at the token level. We want to investigate the feasibility of inducing context-dependent specialization through Sequence-level routing.

3 Experiments

All our experiments are rooted in the GPT2-small (124M)[1] model. The MoE component is added to the feedforward layer (FFN) in *every* transformer block, with an MLP router network deciding which expert(s) to route each sequence/token to. Unlike common practices in larger language models, we drop expert capacity limit in our experiments. We add a load balancing loss with weight ($\lambda = 0.01$) for all our experiments if not specified. See Appendix A.1 for further technical details.

Our experiments were conducted using two different datasets, one is multilingual Wikipedia (Foundation) with documents in English, German, French, and Italian four languages (17B tokens), and the other is OpenWebText (Gokaslan & Cohen, 2019), which is gathered from scrapping URLs from Reddit posts and only available in English (9B tokens). We report our experimental results as the average over the last 100 iterations, to address the inability to conduct multiple runs with different seeds imposed by our computing resources limitation.

When conducting Sequence-level routing, we simply add up the gating outputs of each token and subsequently apply a softmax operation. The Global routing is done by letting all later layers follow the routing outputs of the very first layer, and thus the active gating parameters are only from the very first layer. While there theoretically exists a variance in the number of routing parameters between Global and Sequence-/Token-level routing, the relatively small number of gating parameters compared to other blocks mitigates any substantial disparity in total parameter counts.

3.1 Performance Impact of Design Choices

Note our Global Top-1 model represents the classic MoE structure from the 90s (Jacobs et al., 1991), Token-level Layer-wise Top-1 mimics the switch transformer (Fedus et al., 2022) idea, while Token-

[1] https://github.com/karpathy/nanoGPT

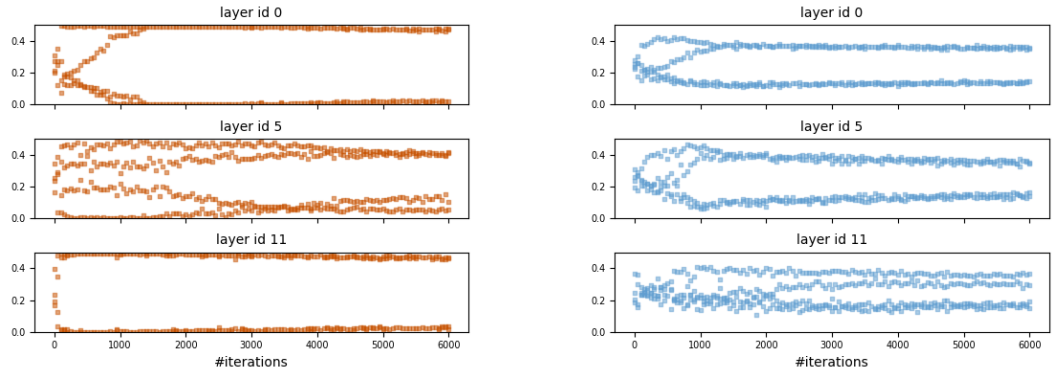

Figure 1: Expert activation frequency during training for Sequence-level routing. Left: pretraining without load balancing loss, resulting in validation perplexity 10.674. Right: with load balancing loss ($\lambda = 0.01$), resulting in validation perplexity 10.667.

level Layer-wise Top-2 routing represents the most popular MoE implementation nowadays (Shazeer et al., 2017; Lepikhin et al., 2020; Jiang et al., 2024).

The baseline methods are selected as the dense model counterparts matching the total number of parameters or the number of active parameters in the forward pass. To do so, we simply duplicate FFN blocks. Examining Table 1, it is evident that with the same amount of total parameters, only the combination of Token-level Layer-wise Top-2 routing surpasses the dense model baselines. Token-level Layer-wise Top-1 routing outperforms the dense model baseline matching the active parameter count by a large margin, while Sequence-level Layer-wise Top-2 routing performs comparably to its dense counterpart with an equivalent number of active parameters. In general, we observe that the routing unit has the biggest impact on validation performance.

3.1.1 Does Expert collapse hurt in Sequence-level routing?

As previously reported in Fedus et al. (2022); Zoph et al. (2022b), expert collapse is always observed without adding an auxiliary loss. Does expert collapse indeed hurt a model's validation performance? For the experimental setup described in Section 3, without an auxiliary loss, we witnessed expert collapse in early and late layers for Sequence-level routing. This is shown in the left panel of Figure 1. Nonetheless, the validation perplexities with or without load balancing loss are almost identical, confirming the findings in Nie et al. (2022); Yang et al. (2021) for Token-level routing. This could indicate that we do not necessarily need the same amount of experts in every layer, as in Du et al. (2022). It should be emphasized that none of our experiments involved the utilization of an expert capacity factor. When an expert capacity factor is applied, imbalanced expert load can potentially result in some tokens not being allocated to any expert, thereby deteriorating validation performance. A more detailed analysis of expert activation is available in Appendix A.4.

3.1.2 Does more experts and more activated experts always help?

We ablate the design choices for the number of activated parameters beyond Top-1 and Top-2, i.e. Top-K routing. In fact, for every sequence or token at each layer, there exist $\binom{N}{K}$ combinations of experts to which activations can be directed, resulting in $12^{\binom{N}{K}}$ distinct path configurations for each routed unit. An increase in the value of N expands the range of routing options for each sequence, while a larger value of K leads to the activation of more neurons within a layer. To investigate the effects, we replicate FFN block N times, indicating N experts. Our findings reveal that *increased activated experts (K) improve performance for sequence-level routing, whereas token-level routing benefits more from a greater number of experts (N).*

(N, K)	**Validation Perplexity** *Sequence*	*Token*
(2 , 2)	10.648	10.761
(4 , 2)	10.667	9.907
(6 , 2)	10.674	**9.467**
(4 , 1)	11.573	10.188
(4 , 3)	**10.107**	9.711

Table 2: The impact of number of experts (N) and number of activated experts (K). Pretrained on Multilingual Wikipedia dataset.

Additionally, routing at a sequence level shows that the $(4, 1)$ combination significantly underperforms compared to $(4, 2)$, likely because it lacks dominant experts, as further elaborated in Section 3.2. In contrast to Token-level routing, we see the most performance gain for Top-1 routing and diminishing return for larger K's. This is consistent with the findings of Clark et al. (2022); Yang et al. (2021). It is noteworthy that the small performance gap between Top-1 and Top-2 routing does not seem to extend to larger models as found by Yang et al. (2021); Lewis et al. (2021), where the expert capacity factor is applied.

3.2 Does expert specialization exist?

3.2.1 What does a router learn?

Frozen routing. To understand if the learned routers (*Learned*) have learned something meaningful, we design experiments where the routing parameters were either frozen at initialization (*Frozen*) or re-initialized randomly (*Random*) at each iteration. It is noteworthy that frozen routing shares a conceptual similarity with hash routing. However, the routing result is embedding dependent instead of token id dependent as in Roller et al. (2021). That is, in our frozen routing setup, expert specialization can still be learned by letting embedding layers adapt to a frozen router head. Layer-wise frozen routing almost gives the same performance as Layer-wise learned routing, as shown in Table 3, supporting this assumption. This finding aligns with Dikkala et al. (2023).

Dataset	**Routing Level**	**Routing Strategy**	**Validation Perplexity**	
			Sequence	*Token*
Multilingual	Global	Language	11.716	-
	Global	Learned	11.605	-
	Layer-wise	Frozen	**10.601**	**9.810**
	Layer-wise	Random	11.685	11.752
	Layer-wise	Learned	**10.666**	**9.907**
OpenWebText	Layer-wise	Frozen	**24.590**	**22.687**
	Layer-wise	Random	26.429	26.879
	Layer-wise	Learned	**24.980**	**22.632**

Table 3: Comparison of different routing strategies.

Language guided routing versus learned Global routing. In the multilingual setup, as languages are naturally different, we examine if our learned router can group languages as expected when routing at a sequence level. One interesting baseline we compared to here is hard-coded per-language routing (*Language*), where we assign the tokens to experts based on the language category instead of a learned gating. To our surprise, this model does not perform as well as expected. One might intuitively anticipate that each expert, assigned a more uniform task (arguably simpler), would learn more rapidly and proficiently. The result could indicate that tokens from a specific language are not clustered in the embedding space. Instead, they are grouped by some other latent features. This seems to align with the findings by Fan et al. (2023) that languages from the same language group can be helpful in learning a low-resource language. Given our learned Global routing gives a roughly similar performance, we conclude that more routing path possibilities are needed, i.e. Layer-wise routing. The learned router also does not show an ability to differentiate languages as discussed in the next Section 3.2.2.

3.2.2 The existence of weak experts?

To validate the presence of expert specialization in Sequence-level routing, we assess our pre-trained model's performance on unseen tasks. We examine whether there is a discernible pattern in expert activation for distinct tasks. Six tasks from the MMLU dataset (Hendrycks et al., 2021) and four languages from the XNLI dataset (Conneau et al., 2018) and X-stance dataset (Vamvas & Sennrich, 2020) were selected to determine if experts acquired domain- or language-specific knowledge.

Regardless of whether the model underwent pre-training on English-only or multilingual datasets, the evaluation of various tasks in English consistently demonstrates similar patterns in expert assignment, depicted in the left and middle plots of Figure 2. For instance, there appears to be a

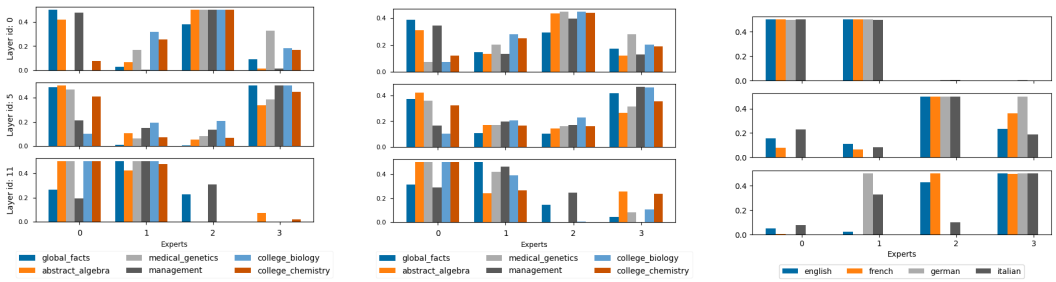

Figure 2: Layer-wise expert assignment results (Sequence-level Top2 routing). From left to right: (1) pretrained on OpenWebText dataset, evaluated on 6 categories of MMLU dataset; (2) pre-trained on Multilingual Wikipedia dataset, evaluated on 6 categories of MMLU dataset; (3) pre-trained on Multilingual Wikipedia dataset, evaluated on 4 languages from XNLI dataset and X-Stance Dataset.

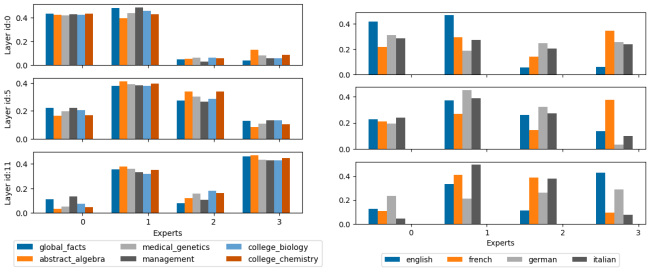

Figure 3: Layer-wise expert assignment results (Token-level Top2 routing) Left: pre-trained on Multilingual Wikipedia dataset, evaluated on 6 categories of MMLU dataset; Right: pre-trained on Multilingual Wikipedia dataset, evaluated on 4 languages from XNLI dataset and X-Stance Dataset.

comparable routing trend for both global facts and management. The uneven distribution of tasks also suggests experts have different specializations in task categories. Expert assignments exhibit language-agnostic characteristics, as showcased in the rightmost panel of Figure 2, hinting at potential benefits in concurrently learning concepts across related languages.

A pre-trained model with Token-level routing shows very different routing behaviors, as shown in Figure 3. Clearly, there is no distinction in experts' ability to handle different tasks, as evidenced by roughly equal assignment of tokens to experts. However, we observe uneven assignments of various languages to different experts, likely stemming from differences in language tokenization.

4 Conclusion

This study delves into the training and design choices of Mixture of Experts (MoEs), focusing on their impact on model performance and expert specialization. Our results demonstrate that sequence-level routing can incorporate context information during routing and foster weak expert specialization related to topics. Yet it does not yield improved performance, as evidenced by validation perplexity. The resulting performance on downstream tasks from sequence-level routing remains an interesting future direction to explore.

In terms of preferred design choices, distinctions arise between routing at the token or sequence level: enhanced activated experts (K) enhance performance for sequence-level routing, while token-level routing benefits from a higher number of experts (N). Ultimately, the conventional Token-level Layer-wise Top2 routing should consistently be favored based on validation perplexity. Practically, we demonstrate that achieving expert specialization during MoE training is attainable by randomly initializing router weights. We hope these observations can contribute to a better understanding of MoE design choices for practitioners.

REFERENCES

Aidan Clark, Diego De Las Casas, Aurelia Guy, Arthur Mensch, Michela Paganini, Jordan Hoffmann, Bogdan Damoc, Blake Hechtman, Trevor Cai, Sebastian Borgeaud, et al. Unified scaling laws for routed language models. In *International Conference on Machine Learning*, pp. 4057–4086. PMLR, 2022.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018.

Damai Dai, Chengqi Deng, Chenggang Zhao, R. X. Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y. Wu, Zhenda Xie, Y. K. Li, Panpan Huang, Fuli Luo, Chong Ruan, Zhifang Sui, and Wenfeng Liang. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models, 2024.

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

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, Barret Zoph, Liam Fedus, Maarten P Bosma, Zongwei Zhou, Tao Wang, Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathleen Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc Le, Yonghui Wu, Zhifeng Chen, and Claire Cui. GLaM: Efficient scaling of language models with mixture-of-experts. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 5547–5569. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/du22c.html.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, et al. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48, 2021.

Simin Fan, Matteo Pagliardini, and Martin Jaggi. Doge: Domain reweighting with generalization estimation. *arXiv preprint arXiv:2310.15393*, 2023.

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

Wikimedia Foundation. Wikimedia downloads. URL https://dumps.wikimedia.org.

Aaron Gokaslan and Vanya Cohen. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019.

Daniel Grittner. nanogpt. https://github.com/danielgrittner/nanoGPT-LoRA, 2023.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models, 2022.

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

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mixtral of experts, 2024.

Andrej Karpathy. nanogpt. `https://github.com/karpathy/nanoGPT/`, 2023.

Sneha Kudugunta, Yanping Huang, Ankur Bapna, Maxim Krikun, Dmitry Lepikhin, Thang Luong, and Orhan Firat. Exploring routing strategies for multilingual mixture-of-experts models, 2021. URL `https://openreview.net/forum?id=ey1XXNzcIZS`.

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

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

Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A. Smith, and Luke Zettlemoyer. Branch-train-merge: Embarrassingly parallel training of expert language models, 2022.

Xiaonan Nie, Shijie Cao, Xupeng Miao, Lingxiao Ma, Jilong Xue, Youshan Miao, Zichao Yang, Zhi Yang, and Bin CUI. Dense-to-sparse gate for mixture-of-experts, 2022. URL `https://openreview.net/forum?id=_4D8IVs7yO8`.

Hai Pham, Young Jin Kim, Subhabrata Mukherjee, David P. Woodruff, Barnabas Poczos, and Hany Hassan Awadalla. Task-based moe for multitask multilingual machine translation, 2023.

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

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

Jannis Vamvas and Rico Sennrich. X-stance: A multilingual multi-target dataset for stance detection, 2020.

Fuzhao Xue, Zian Zheng, Yao Fu, Jinjie Ni, Zangwei Zheng, Wangchunshu Zhou, and Yang You. Openmoe: An early effort on open mixture-of-experts language models. *preprint*, 2023.

An Yang, Junyang Lin, Rui Men, Chang Zhou, Le Jiang, Xianyan Jia, Ang Wang, Jie Zhang, Jiamang Wang, Yong Li, et al. M6-t: Exploring sparse expert models and beyond. *arXiv preprint arXiv:2105.15082*, 2021.

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

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

A Appendix

A.1 Experimental Details

We run our experiments based on the publicly available LoRA extension (Grittner, 2023) of the nanoGPT library (Karpathy, 2023) on a single A100-SXM4-40GB GPU. We use the default setup, but with dropout 0.2, learning rate 9.6e-4, minimum learning rate 9.6e-5, weight decay 0.5 and enabled biases for 6K iterations. Every iteration 1'048'576 tokens are passed through the network (gradient accumulation 128, batch size 8, sequence length 1024), resulting in roughly 6B tokens being seen by the model. For our default set up with 4 experts this follows the Chinchilla scaling law from Hoffmann et al. (2022) to train on approximately 20 tokens per parameter.

For both datasets, we use the OpenAI GPT-2 tokenizer, which gives a slight disadvantage to the non-English languages. However, we observe that the model can still learn the non-English languages, in close language groups for next token prediction.

Assume gating network W_g produces logits $h(x) = W_g x$. Expert blocks are denoted as E_is. Our Top-1 gating follows the implementation of switch transformer (Fedus et al., 2022).

$$p_i(x) = \frac{e^{h_i(x)}}{\sum_j e^{h_j(x)}} \qquad y = \sum_{i \in \mathcal{T}} p_i(x) E_i(x) \tag{1}$$

Top-2 gating implementation follows the implementation of Jiang et al. (2024), except for Sequence-Level routing we apply the softmax operation twice, as shown below:

$$p_i(x) = \frac{e^{h_i(x)}}{\sum_j e^{h_j(x)}} \qquad y = \sum_{i \in \mathcal{T}} \frac{e^{p_i(x)}}{\sum_{j \in \mathcal{T}} e^{p_j(x)}} E_i(x) \tag{2}$$

Note $\mathcal{T}$ in the formulas denote the set of Top-K index/indices.

A.2 On Router Choices

Empirically, using one-layer MLP or two-layer MLP as the router network does not make a difference with regard to validation performances, as shown in Figure 4.

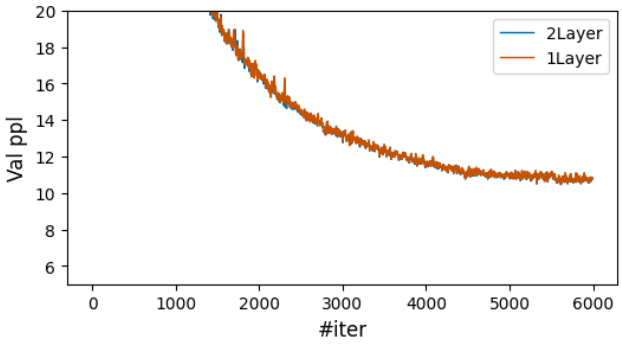

Figure 4: Validation perplexity versus training iterations.

A.3 Expert Assignments

We visualize expert assignments when routing with our pre-trained model (Top-2 Layer-wise Sequence-level routing) on OpenWebtext data in Figure 5. Frozen routing and learned routing gave similar routing results, indicating weak expert specialization in topics. In the middle panel of Figure 5, we notice the co-existence of a common expert and weak specialized experts, aligning the design choice of shared experts in Dai et al. (2024).

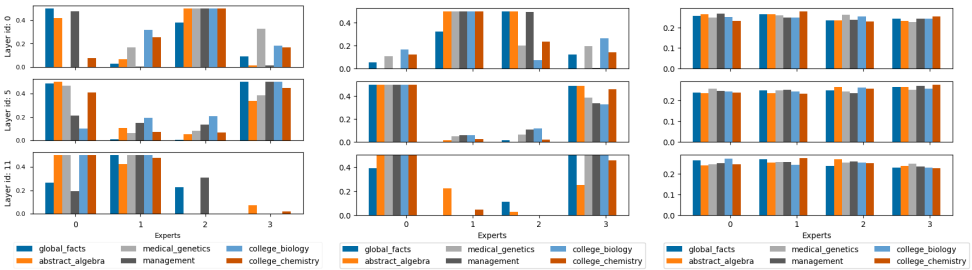

Figure 5: Expert assignment results when evaluating on MMLU dataset using different pre-trained models on OpenWebText dataset. From left to right: (1) learned routing; (2) frozen routing; (3) random routing (Sequence-level routing is performed).

A visualization of expert assignments when routing with our pre-trained model (Top-2 Layer-wise Token-level routing) on Multilingual Wikipedia dataset is shown in Figure 3. In contrast to weak topic specialization (Left panel of Figure 2) when routing at a sequence level, Token-level routing does not bring any expert specialization in topic domains, as suggested in the left plot of Figure 3. The almost even expert assignment indicates the lack of expert specialization, similar to the observations in random routing illustrated in Figure 5. The main difference arises from the balanced expert load in random routing, as opposed to the uneven distributions found in learned routing.

A.4 Layer wise expert activation

We visualize Layer-wise expert activation when routing at a sequence level without load balancing loss in Figure 6. When we refer to expert activation, we are describing the frequency at which each expert is activated. For each iteration (x-axis), there should be $N = 4$ points denoting the corresponding activated frequency of each expert. We observe the following interesting aspects:

- Experts exhibit the ability to recover from collapsing.
- Early and late layers demonstrate a tendency to experience more expert collapse.

To provide readers with a clear visualization of layer-wise expert activations under the application of load balancing loss (with λ = 0.01), we additionally plot the Top-2 and Top-3 routing results respectively in Figure 7 and Figure 8.

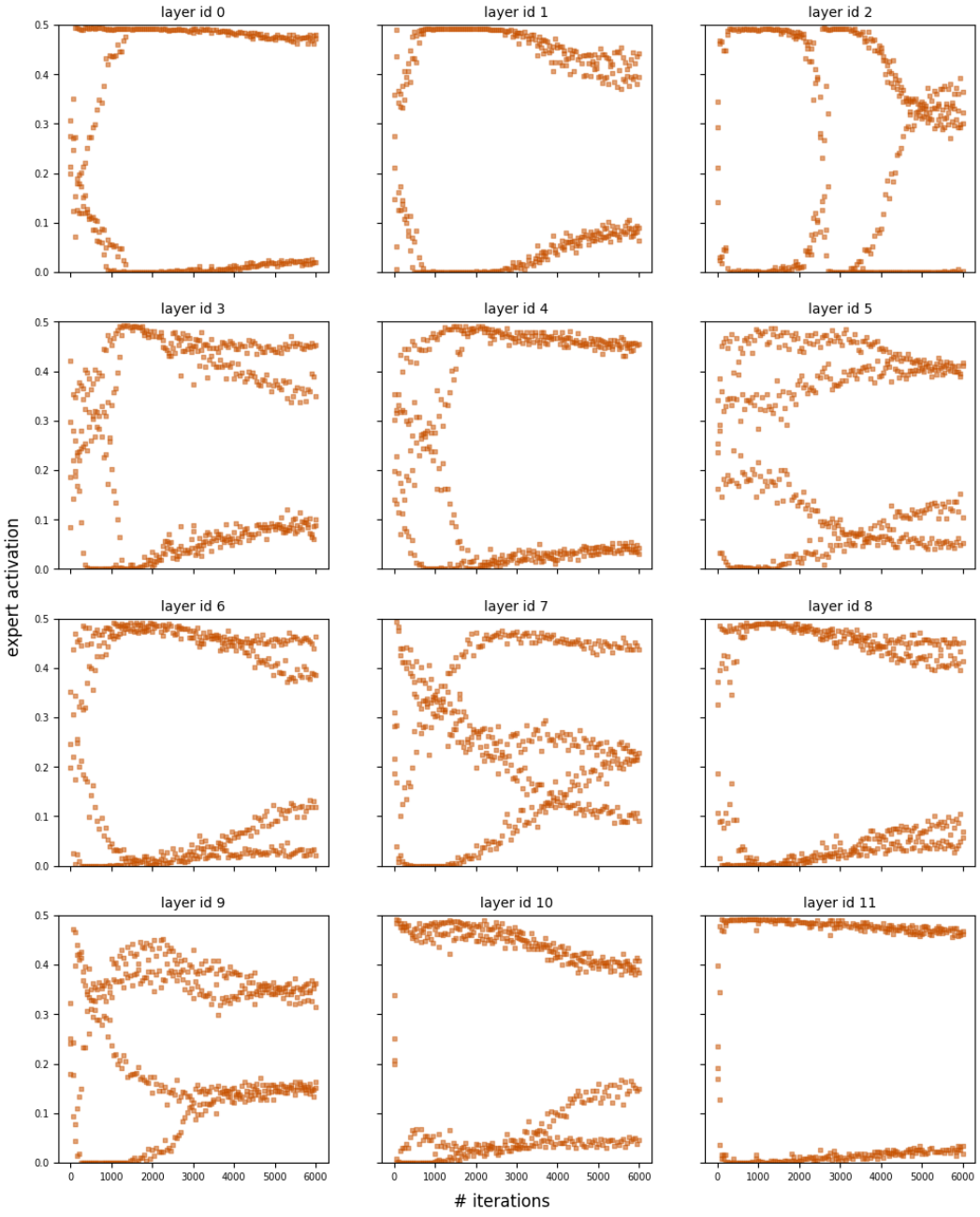

Figure 6: Expert activations from Layer-wise Sequence-level Top-2 routing when no load balancing loss is applied.

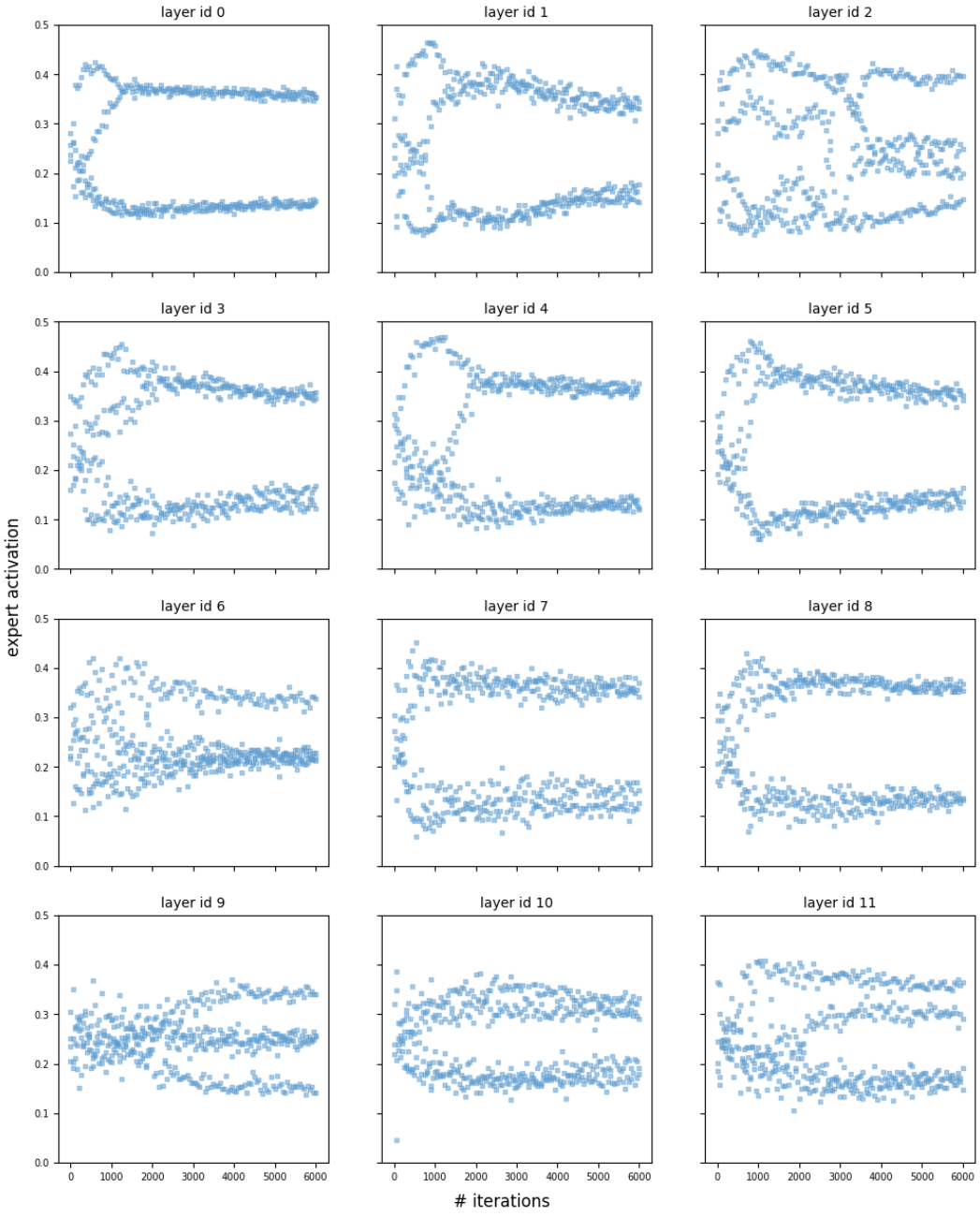

Figure 7: Expert activations from Layer-wise Sequence-level Top-2 routing when load balancing loss is applied.

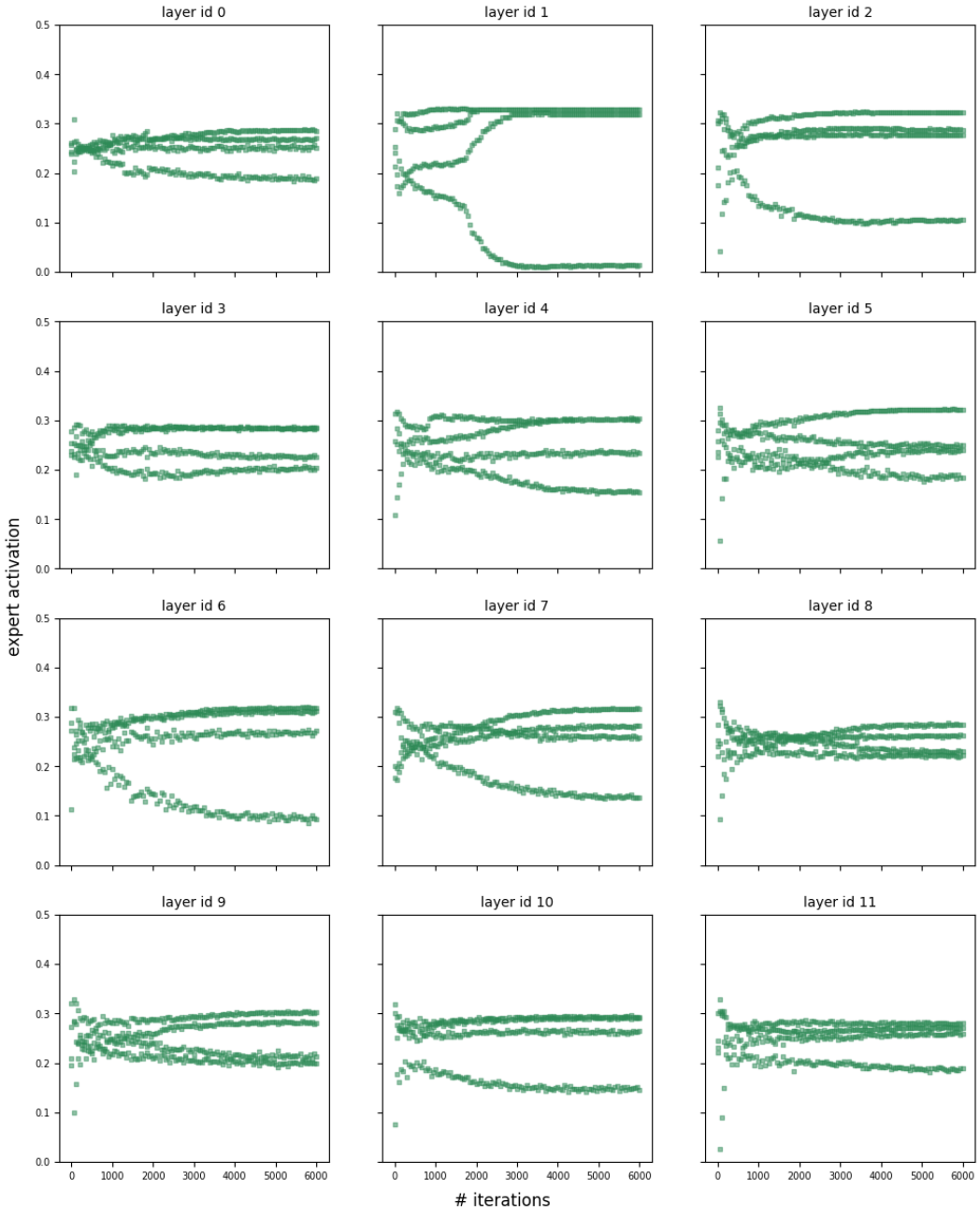

Figure 8: Expert activations from Layer-wise Sequence-level Top-3 routing when load balancing loss is applied.