μ-Parametrization for Mixture of Experts

Anonymous Authors[1]

Abstract

Recent years have seen a growing interest and adoption of LLMs, with μTransfer becoming a key technique for tuning hyperparameters in large-scale training. Meanwhile, Mixture-of-Experts (MoE) has emerged as a leading architecture in extremely large models. However, the intersection of these two advancements has remained unexplored. In this work, we derive a μ-Parameterization (μP) for MoE, providing theoretical guarantees for feature learning across model widths in both the router and experts. We empirically validate our parameterization and further investigate how scaling the number of experts and granularity affects the optimal learning rate.

1. Introduction

Scaling deep learning models has become a fundamental driver of progress in modern AI (Fedus et al., 2022; Touvron et al., 2023; OpenAI, 2025). Nevertheless, efficiently tuning hyperparameters across different model sizes remains a major challenge, often requiring extensive trial-and-error or computationally expensive grid searches (Feurer et al., 2015). The μParameterization (μP) (Yang, 2021) offers a principled solution by enabling stable and predictable training dynamics across model widths—without the need to retune learning rates or initialization schemes. By reparameterizing models to preserve feature learning in the infinite-width limit, μP makes it possible to identify optimal hyperparameters on small models and seamlessly transfer them to larger ones, significantly reducing tuning costs.

On the other hand, Mixture-of-Experts (MoE) models have emerged as a compelling approach for scaling large language models efficiently, offering substantial computational savings through sparse activation (Clark et al., 2022; Ludziejewski et al., 2024). However, the sparsity patterns and routing mechanisms intrinsic to MoE architectures fall outside the scope of current μP theory. Consequently, it remains unclear whether the parameterization developed for dense Transformers can be directly applied to MoE models, or whether adaptations are required to retain the transferability and stability benefits that μP provides.

In this work, we extend μP to Mixture-of-Experts (MoE) architectures, offering both theoretical grounding and empirical validation. Our main contributions are:

- **Theoretical framework for μP in MoE.** Building on (Yang et al., 2022), we derive a parameterization scheme that ensures that feature learning is preserved across all weights within MoE layers.
- **Empirical validation of learning rate transfer.** We show that our parameterization enables consistent learning dynamics across model widths, confirming that hyperparameters can be transferred effectively in MoE setups.
- **Investigation of hyperparameter transferability across other MoE parameters.** We observe that learning rate transferability breaks down across different values of top-k and granular expert sizes, highlighting a boundary of current hyperparameter transferability in MoE settings.

2. Background and Related Work

Mixture of Experts. Mixture of Experts was originally introduced by (Jacobs et al., 1991), and later proposed in the context of language modeling by (Shazeer et al., 2017). This approach has since been succesfully integrated into the Transformer (Vaswani et al., 2023) architecture in multiple works, including (Fedus et al., 2022; Lepikhin et al., 2020; Du et al., 2022; Zhou et al., 2022). +In a Transformer model, the MoE layer is typically constructed by replacing the Feed-Forward component with a set of *experts*. Each expert retains the design of the original Feed-Forward layer, consisting of two linear layers with a non-linearity between them. Crucially, for any given input token, only a fraction of these experts are activated. The selection of experts for each token is determined by a routing mechanism - a simple linear layer followed by a softmax normalization and a Top-k choice.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

In the standard, Switch layer (Fedus et al., 2022), each of the experts is of the same size as in the corresponding dense (non-MoE) Transformer. This assumption is relaxed in fine-grained MoE (Dai et al., 2024; Ludziejewski et al., 2024), where for granularity G, the hidden size of each expert is reduced by a factor of G, while the number of experts and the router's top-k value are both multiplied by G. This way, the model has greater flexibility in mapping tokens to experts, while the number total and activated parameters remains approximately constant.

Zero-shot hyperparameter transfer. Standard parametrization (SP) often fails to preserve stability and hyperparameter transfer in scaling neural networks. To overcome this limitation, (Yang, 2021) introduced Maximal Update Parametrization (μParametrization, or μP). μP ensures that each layer in a network receives updates of the same order of magnitude during training, regardless of width. This allows for what is known as the feature learning regime, where internal representations evolve in a meaningful way as training progresses. Crucially, μP enables hyperparameter transfer across model sizes: one can tune learning rates and initialization on a small model and zero-shot transfer them to a large one, as shown empirically and theoretically in (Yang et al., 2022). This paradigm, called μTransfer, has been shown to dramatically reduce the cost of training large models while maintaining performance. Later works (Yang et al., 2024; Everett et al., 2024) reformulate and generalize μP theory, while (Dey et al., 2025) and (Yang et al., 2023) include transfer across model depths. Despite its success on many architectures such as Transformers and ResNets, extending μP to Mixture-of-Experts models remains an open challenge. In this work, we address this open problem.

3. Principled approach to μP for MoE

In this section, we analyze the behavior of MoE (Fedus et al., 2022) during training. We derive a parameterization that ensures feature learning in all of its weights across different model widths.

3.1. Intuition

In Tensor Programs 5 (TP5) (Yang et al., 2022), weight matrices are categorized based on the dimensions they connect. A weight is referred to as a *hidden weight* if it maps from an infinitely wide layer to another infinitely wide layer, as is typical within the internal blocks of deep networks. In contrast, an *output weight* maps from an infinitely wide layer to a layer of fixed finite size, such as a classification head. This structural distinction determines how the weight should be scaled at initialization and during optimization. Hidden weights receive gradient updates of magnitude $\Theta(1/n)$, where n denotes the layer width. Output weights, by contrast, receive gradients of constant order $\Theta(1)$. Differentiation between those weights ensures stable training and enables zero-shot hyperparameter transfer across model scales, a core goal of the μTransfer paradigm (Yang et al., 2022).

Using convention from TP5 expert weights map infinite dimension to infinite dimension, so they should act as "hidden weights", and router weight maps infinite to finite dimension, so it should act as "output weight". We will now verify those intuitions.

3.2. Definitions and notation

We model the MoE layer in the form of a Switch Transformer (Fedus et al., 2022). It consists of:

- A router matrix $R \in \mathbb{R}^{n_{\text{experts}} \times n}$,
- Two layers of an expert E:
  $E_1 \in \mathbb{R}^{n_{\text{experts}} \times 4n \times n}, \quad E_2 \in \mathbb{R}^{n_{\text{experts}} \times n \times 4n}$.

where by n we denote the width of the model that gets scaled to infinity.

The forward pass of the MoE layer is computed as:

$$\begin{aligned} E(x) &= E_2 \mathrm{ReLU}(E_1 x), \\ R(x) &= \mathrm{softmax}(\text{top-k}(Rx)), \\ \mathrm{MoE}(x) &= E(x)^T R(x). \end{aligned}$$

We now derive the correct μParameterization (μP) for Mixture-of-Experts (MoE) architectures by adapting the principles established in Tensor Programs V (TP5) (Yang et al., 2022). Our goal is to ensure that all relevant quantities (activations, gradients, and updates) scale in a way that allows for stable and predictable training dynamics across width, enabling hyperparameter transfer.

3.3. Desiderata

Following TP5, we define the key properties that a correctly μ-parametrized model must satisfy:

1. At initialization, all hidden representations $h(x)$ in the network should scale as $\Theta(1)$.
2. The model output logits $f(x)$ should be $O(1)$ at initialization.
3. After one optimization step, the changes to hidden representations $\Delta h(x)$ and output logits $\Delta f(x)$ should be $\Theta(1)$.

These desiderata ensure that no matter how wide the model is, the output of each layer and the logits stay constant in size and so the feature learning (Yang et al., 2022) occurs.

	Embedding	Unembedding	Attention (Q, K, V, O)	Feed-forward (dense)	Experts (MoE)	Router (MoE)
Init. Var.	1.0	1.0	1/fan_in	1/fan_in	1/fan_in	1/fan_in | 1.0
Multiplier	1.0	1/fan_in	1.0	1.0	1.0	1.0 | 1/fan_in
LR (Adam)	1.0	1.0	1/fan_in	1/fan_in	1/fan_in | 1/fan_in	1.0

Table 1: The table presents parameterizations of dense and MoE Transformers, showing parameter scaling in big-O notation. Dense transformer μP is indicated in blue. MoE parameterizations build on dense μP. *sim*P MoE is marked in red, while the theoretically grounded μP MoE is shown in green.

By "vector $v \in \mathbb{R}^n$ is $\Theta(n^a)$", we mean $\frac{||v||^2}{n} = \Theta(n^{2a})$ where $||\cdot||$ is the standard Euclidean norm (same as in (Yang et al., 2022)). Intuitively, this means that the typical entry of v is of size $\Theta(n^a)$. We use the same definition for vector v being $O(n^a)$ and $\Omega(n^a)$, and we use similar definitions for matrices to mean "typical entry size". We will assume that all layers other than MoE layers follow parametrization from TP5.

3.4. Derivation

3.4.1. Initialization

If we initialize E_1, E_2, R according to TP5, we get:

- $E(x) = \Theta(1)$
- $Rx = O(1)$
- $R(x) = \Theta(1)$
- MoE $(x) = \Theta(1)$

Only non-standard part is Rx, but it has no direct effect on the size of the output of the layer, which stays $\Theta(1)$.

3.4.2. Optimizer step

In TP5, gradients with respect to hidden activations are typically $\Theta(1/n)$, and gradients with respect to output layer (pre-)activations are $\Theta(1)$. We verify that the MoE components obey the same behavior.

We assume the same gradient norms as in TP5, $\nabla h = \Theta(1/n)$ for hidden layers, and $\nabla h = \Theta(1)$ for output layer. In the case of MoE, this means $\nabla \text{MoE}(x) = \Theta(1/n)$. Thus we obtain:

$$\begin{aligned} \nabla E(x) &= R(x) \nabla \text{MoE}(x)^T \\ &= \Theta(1) \cdot \Theta(1/n) = \Theta(1/n) \end{aligned} \tag{1}$$

Since $R(x), \text{MoE}\ (x)$ are 1-dimensional vectors the entry size is simply product of entry sizes. Then:

$$\begin{aligned} \nabla R(x) &= E(x) \nabla \text{MoE}(x) \\ &= \Theta(1) \cdot \Theta(1/n) \cdot n = \Theta(1) \end{aligned} \tag{2}$$

This equality is less obvious and requires proof that $E(x)$ and $\nabla \text{MoE}\ (x)$ are correlated. We prove that correlation in Appendix B. For correlated vectors $v, u \in \mathbb{R}^n$ quantity $v^T u$ has expected size $\Theta(v)\Theta(u) \cdot \text{corr}(v, u) \cdot n$, which follows from the Law of Large Numbers.

For $R(x)$, since Rx is $O(1)$ and we take top-k over constant k, those operations do not change size of the gradient and gradient over Rx is still $\Theta(1)$, and $\nabla E_1 x, \nabla E_2 \text{ReLU}(E_1 x)$ are $\Theta(1/n)$ since they mimic standard MLP layers. That means E_1, E_2, R receive the same gradient sizes as hidden weights and output weight, respectively. This shows that they behave in the same way in training as their respective weight types from TP5, which shows that our intuition is correct and E_1, E_2 should be scaled as hidden weights, while R should be scaled as output weight.

4. Experimental results and alternative views on hyperparameter transfer in MoE

This section presents experimental results on learning rate transfer in MoE Transformers, providing empirical validation of our μP-based parameterization for MoE models.

4.1. Parameterizations for MoE

In Section 3 we develop theory for μP for MoE where we re-parametrize both router and experts. In the experimental validation, we also include a simplified parameterization where each expert is treated as a dense MLP without modifying the router, an approach we term *simple*-Parameterization MoE, or *sim*P-MoE. *sim*P-MoE follows the intuitions from μP in dense Transformers, leveraging the structural similarity between each expert and a Transformer's MLP block. The details of both parameterizations are summarized in Table 1.

4.2. Model width:

We conduct experiments verifying transferability of learning rate with respect to the model width (see Figure 1). Both μP-MoE and *sim*P-MoE achieve learning rate transfer. In

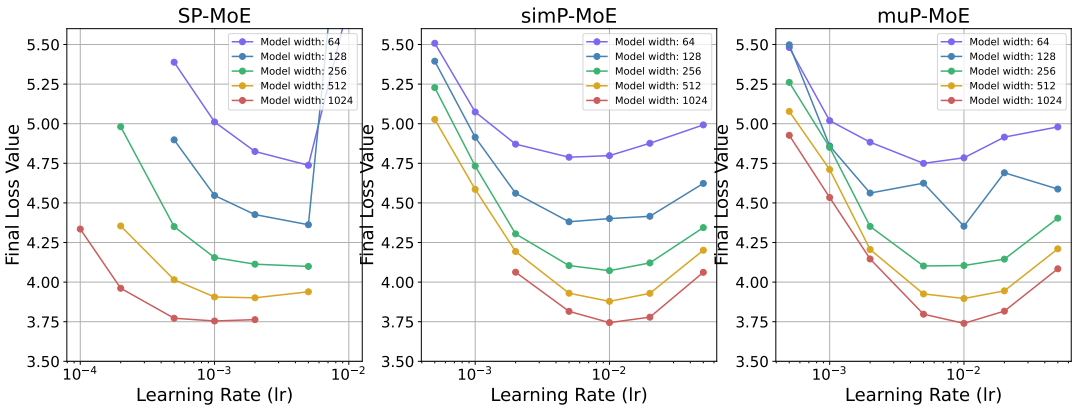

Figure 1: The plots present MoE performance for varying learning rates in the following set-ups: standard parametrization (SP) with no scaling on the left. *sim*P - treating each Expert like a FeedForward layer in the middle. μP - our theory applied to MoE layer on the right. While in the case of SP, the optimal learning rate is different for different model sizes, both reparameterizations achieve learning rate transfer across model widths.

both parameterizations the optimal learning rate seems to shift a little, so that wider models have a slightly higher optimal value. This result is similar to the original TP5 and to our experiments on dense models (Figure **??** in Appendix). This may be due to instabilities of architectures with a large depth-to-width ratio, although a proper investigation would be an interesting future work direction. Two training instances of μP MoE with an embedding dimension of 128 diverged, suggesting that μP MoE may be less stable than *sim*P. However, this observation is not conclusive and would require further experiments to validate.

4.3. Scaling other MoE dimensions:

In the previous section we have shown μ-Parametrization for MoE for varying model width. In this section, we investigate whether two other parameters of MoE architecture necessitate respective reparameterization. The first one is the number of experts that describes MoE model's size. This parameter increases the model performance without increasing the computational cost, but sacrifices memory footprint (Clark et al., 2022; Ludziejewski et al., 2025). The other parameter is granularity as defined in Sec. 2. This parameter enables control over the expert size, keeping computational cost fixed. It can be used to adjust expert's size to the available hardware like in (Dai et al., 2024).

Figure 2(a) shows learning rate grid searches for varying numbers of experts. The experiments confirm that having more experts leads to lower final loss. We find that the optimal learning rate is stable. In Figure 2(b), we show MoE performance for different learning rates and granularities. Contrary to the number of experts, optimal learning rate does not directly transfer between granularities. This is most likely the result of top-k adjusted by the granularity factor,

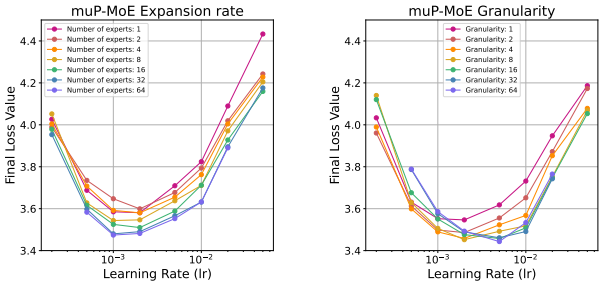

Figure 2: (a) Varying the number of experts. Given our muP parametrization the optimal learning rate is preserved across varied number of experts. (b) Varying granularity. Learning rate is not preserved across different granularities.

or decreasing hidden dimension of each expert, as scaling the number of experts alone does not change the optimal learning rate. It is important to keep in mind that both ablations break the assumption of constant router output dimensions, which would call for adjustment of our theory. We leave the investigation of these results for future work.

5. Conclusions

In this work, we derive a μ-Parameterization for MoE Transformer models, enabling hyperparameter transfer across model width. We empirically validate our theoretical findings. Additionally, we explore two other MoE scaling strategies and observe that scaling the top-k or the expert hidden dimension disrupts learning rate transferability, while scaling the number of experts preserves it. Investigating granularity scaling remains a direction for future work.

References

Clark, A., De Las Casas, D., Guy, A., Mensch, A., Paganini, M., Hoffmann, J., Damoc, B., Hechtman, B., Cai, T., Borgeaud, S., Van Den Driessche, G. B., Rutherford, E., Hennigan, T., Johnson, M. J., Cassirer, A., Jones, C., Buchatskaya, E., Budden, D., Sifre, L., Osindero, S., Vinyals, O., Ranzato, M., Rae, J., Elsen, E., Kavukcuoglu, K., and Simonyan, K. Unified scaling laws for routed language models. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), Proceedings of the 39th International Conference on Machine Learning, volume 162 of Proceedings of Machine Learning Research, pp. 4057–4086. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/clark22a.html.

Dai, D., Deng, C., Zhao, C., Xu, R. X., Gao, H., Chen, D., Li, J., Zeng, W., Yu, X., Wu, Y., Xie, Z., Li, Y. K., Huang, P., Luo, F., Ruan, C., Sui, Z., and Liang, W. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models, 2024.

Dey, N., Zhang, B. C., Noci, L., Li, M., Bordelon, B., Bergsma, S., Pehlevan, C., Hanin, B., and Hestness, J. Don't be lazy: Completep enables compute-efficient deep transformers, 2025. URL https://arxiv.org/abs/2505.01618.

Du, N., Huang, Y., Dai, A. M., Tong, S., Lepikhin, D., Xu, Y., Krikun, M., Zhou, Y., Yu, A. W., Firat, O., Zoph, B., Fedus, L., Bosma, M., Zhou, Z., Wang, T., Wang, Y. E., Webster, K., Pellat, M., Robinson, K., Meier-Hellstern, K., Duke, T., Dixon, L., Zhang, K., Le, Q. V., Wu, Y., Chen, Z., and Cui, C. Glam: Efficient scaling of language models with mixture-of-experts, 2022.

Everett, K., Xiao, L., Wortsman, M., Alemi, A. A., Novak, R., Liu, P. J., Gur, I., Sohl-Dickstein, J., Kaelbling, L. P., Lee, J., and Pennington, J. Scaling exponents across parameterizations and optimizers, 2024. URL https://arxiv.org/abs/2407.05872.

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.

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., and Hutter, F. Efficient and robust automated machine learning. In Advances in Neural Information Processing Systems (NeurIPS), pp. 2962–2970, 2015.

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. doi: 10.1162/neco.1991.3.1.79.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), Proceedings of the 17th International Conference on Machine Learning (ICML 2000), pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Lepikhin, D., Lee, H., Xu, Y., Chen, D., Firat, O., Huang, Y., Krikun, M., Shazeer, N., Sepassi, R., Tucker, P., and Zhou, C. Gshard: Scaling giant models with conditional computation and automatic sharding. In Proceedings of the 37th International Conference on Machine Learning (ICML), 2020. URL https://arxiv.org/abs/2006.16668.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization, 2019.

Ludziejewski, J., Krajewski, J., Adamczewski, K., Pióro, M., Krutul, M., Antoniak, S., Ciebiera, K., Król, K., Odrzygóźdź, T., Sankowski, P., Cygan, M., and Jaszczur, S. Scaling laws for fine-grained mixture of experts. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F. (eds.), Proceedings of the 41st International Conference on Machine Learning, volume 235 of Proceedings of Machine Learning Research, pp. 33270–33288. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/ludziejewski24a.html.

Ludziejewski, J., Pióro, M., Krajewski, J., Stefaniak, M., Krutul, M., Małaśnicki, J., Cygan, M., Sankowski, P., Adamczewski, K., Miłoś, P., and Jaszczur, S. Joint moe scaling laws: Mixture of experts can be memory efficient, 2025. URL https://arxiv.org/abs/2502.05172.

OpenAI. Gpt-4o technical report. arXiv preprint arXiv:2410.21276, 2025.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving language understanding by generative pre-training. 2018.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. In Proceedings of the 37th International Conference on Machine Learning, pp. 5541–5551. PMLR, 2020. URL https://arxiv.org/abs/1910.10683.

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, 2017.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E.,

Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. Llama: Open and efficient foundation language models, 2023.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need, 2023.

Yang, G. Tp4: Feature learning in infinite-width neural networks. In International Conference on Learning Representations (ICLR), 2021. URL https://openreview.net/forum?id=K19h3z-4Z.

Yang, G., Hu, E. J., et al. Tp5: Tuning large neural networks via zero-shot hyperparameter transfer. arXiv preprint arXiv:2203.03466, 2022. URL https://arxiv.org/abs/2203.03466.

Yang, G., Yu, D., Zhu, C., and Hayou, S. Tensor programs vi: Feature learning in infinite-depth neural networks, 2023. URL https://arxiv.org/abs/2310.02244.

Yang, G., Simon, J. B., and Bernstein, J. A spectral condition for feature learning, 2024. URL https://arxiv.org/abs/2310.17813.

Zhou, Y., Lei, T., Liu, H., Du, N., Huang, Y., Zhao, V., Dai, A., Chen, Z., Le, Q., and Laudon, J. Mixture-of-experts with expert choice routing, 2022.

Zoph, B., Bello, I., Kumar, S., Du, N., Huang, Y., Dean, J., Shazeer, N., and Fedus, W. St-moe: Designing stable and transferable sparse expert models. arXiv preprint arXiv:2202.08906, 2022.

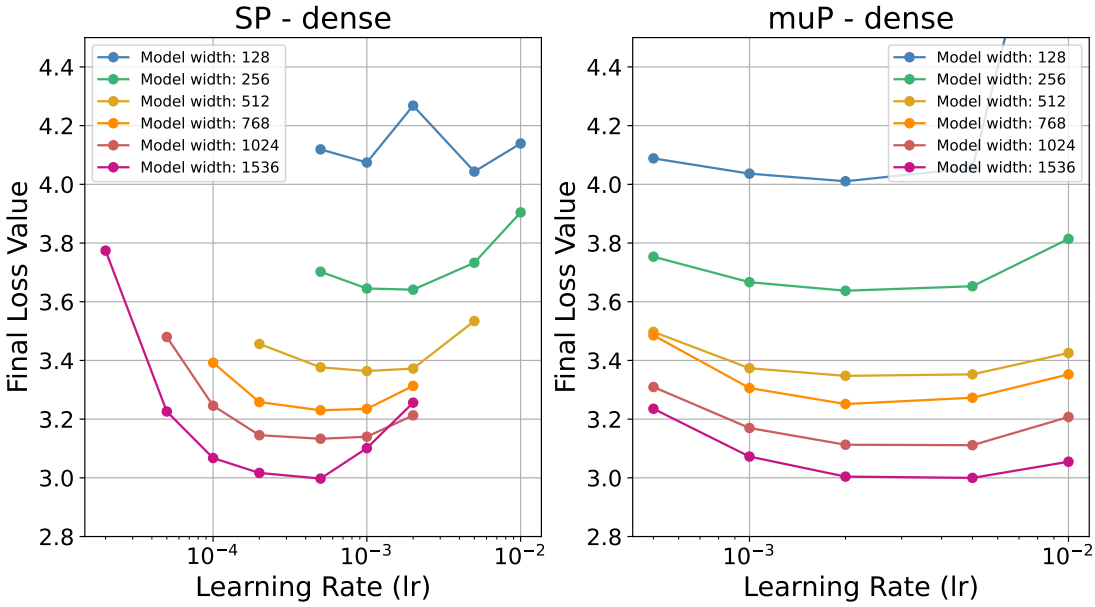

Figure 3: This figure shows our experiments on learning rate transfer in *dense* models. Standard Parameterization (SP) on the lft has different optimal learning rate for each model width, while μP has stable optimum.

A. MuP for Dense Transformer

In this section we verify the findings from (Yang et al., 2022) by implementing the μP for dense models. As opposed to standard parametrization, in the case of muP reparametrization, the optimal learning rates transfer between different model widths.

B. Expert–gradient covariance lemma

We now formalize the intuition that in a µP-parametrized Switch-MoE block, each active expert's forward activation correlates with its backward gradient at order $\Theta(1/n)$, and that the router's gradient norm remains $\Theta(1)$ both immediately after initialization and again after one µ-SGD/Adam update.

Lemma B.1 (Expert–gradient covariance)**.** *Let an L-block Switch-MoE be µP-parametrized with width $n \to \infty$, a fixed number of experts $n_{\text{experts}} = O(1)$, and fixed top-$k = O(1)$. For each block ℓ, define*

$$y^{(\ell)} = \sum_{e=1}^{n_{\text{experts}}} R_e^{(\ell)}\big(x^{(\ell)}\big)\, E_e^{(\ell)}\big(x^{(\ell)}\big), \qquad \delta^{(\ell)} = \nabla_{y^{(\ell)}} L \in \mathbb{R}^n. \tag{3}$$

Assume the inductive hypothesis

$$\frac{1}{n}\sum_{j=1}^{n}\big(\delta_j^{(\ell)}\big)^2 = \Theta(n^{-2}) \implies \delta_j^{(\ell)} = \Theta(n^{-1}) \quad \textit{for typical } j. \tag{H_ℓ}$$

Then for every block ℓ and any active expert e, at both

$$t = 0: \ \textit{immediately after initialization}, \qquad t = 1: \ \textit{after one } \mu\textit{-SGD/Adam step},$$

we have

$$\begin{aligned} \mathrm{Cov}\big(E_{e,j}^{(\ell)},\, \delta_j^{(\ell)}\big) &= \Theta(n^{-1}), \\ \big\|\nabla_{r^{(\ell)}} L\big\|_2 &= \Theta(1). \end{aligned} \tag{4}$$

Proof. **Notation.** For block ℓ, set

$$\begin{aligned} \mathbf{e}_e &= E_{\ell,e}^{(2)}\,\mathrm{ReLU}\big(E_{\ell,e}^{(1)} x^{(\ell)}\big) \in \mathbb{R}^n, \\ R_e &= R_e^{(\ell)}\big(x^{(\ell)}\big) \in \mathbb{R}, \\ \delta &= \delta^{(\ell)} \in \mathbb{R}^n, \\ J^{(\ell)} &= \frac{\partial\, y^{(\ell)}}{\partial\, x^{(\ell)}} \in \mathbb{R}^{n\times n}, \end{aligned} \tag{5}$$

so that $\delta^{(\ell-1)} = (J^{(\ell)})^\top\,\delta$ and $J_{ij}^{(\ell)} \sim \mathcal{N}(0, 1/n)$ under µP.

Step 1: Stein's lemma (holds at $t = 0$ and $t = 1$). Fix expert e, coordinate j, and define

$$\begin{aligned} Z &= \mathbf{e}_{e,j} \sim \mathcal{N}(0, \sigma^2), \\ c &= \sum_{e'\neq e} R_{e'}\,\mathbf{e}_{e',j}, \\ g(z) &= [L'(y^{(\ell)})]_j, \\ y_j^{(\ell)} &= R_e\, z + c. \end{aligned} \tag{6}$$

Then $g'(z) = R_e\,[L''(y^{(\ell)})]_j$ and by (H_ℓ), $[L''(y^{(\ell)})]_j = \Theta(n^{-1})$. Thus

$$\begin{aligned} \mathrm{Cov}(Z, g(Z)) &= \sigma^2\,\mathbb{E}\big[g'(Z)\big] \\ &= \sigma^2\,R_e\,\mathbb{E}\big[L''(y^{(\ell)})\big]_j \\ &= \Theta(1)\cdot\Theta(1)\cdot\Theta(n^{-1}) = \Theta(n^{-1}). \end{aligned} \tag{7}$$

Remark. Because by induction the block-$(\ell+1)$ Hessian entries already scale like $\Theta(n^{-1})$, and passing any such matrix back through a µP-linear layer (whose weights are $\mathcal{N}(0, 1/n)$) multiplies each term by another $1/n$ but sums over n of them, the net effect is still $\Theta(n^{-1})$. In other words, a $1/n$ factor per weight-matrix multiplication exactly preserves the $\Theta(n^{-1})$ scale of $\big[L''(y^{(\ell)})\big]_j$.

Step 2: Router-gradient norm (holds at $t = 0$ and $t = 1$). Summing the covariances over j,

$$\mathbb{E}\big[\mathbf{e}_e^\top\delta\big] = \sum_{j=1}^{n}\mathrm{Cov}(\mathbf{e}_{e,j},\,\delta_j) = n\cdot\Theta(n^{-1}) = \Theta(1). \tag{8}$$

Since $\mathrm{Var}(\mathbf{e}_e^\top\delta) = O(n^{-1})$, Chebyshev's inequality gives $\mathbf{e}_e^\top\delta = \Theta(1)$ with high probability, i.e. the second line of (4).

Step 3: One-step update. Under µ-SGD/Adam with LR η/n on experts and η on router, the factors in (7) and (8) change by at most a $(1 + O(n^{-1}))$ factor, so both lines of (4) hold at $t = 1$.

Step 4: Depth induction. Using $\delta^{(\ell-1)} = (J^{(\ell)})^\top\delta^{(\ell)}$ and $J_{ij}^{(\ell)} \sim \mathcal{N}(0, 1/n)$, one shows $\frac{1}{n}\sum_j(\delta_j^{(\ell-1)})^2 = \Theta(n^{-2})$, establishing $(H_{\ell-1})$. The base case $\ell = L$ is given by Tensor-Programs V; induction completes the proof. □

C. Experimental setup

All models in this study are decoder-only Transformers trained on the C4 dataset (Raffel et al., 2020). We use the GPT-2 tokenizer (Radford et al., 2018) and optimize with AdamW (Loshchilov & Hutter, 2019). Training follows a cosine decay schedule with linear warmup for the first 1% of steps. Weights are initialized with a normal distribution, as the theory of

Tensor Programs assumes (Yang et al., 2022). Mixed precision training is applied, with Attention component computed at high precision. The models employ MLP with ReLU activations. MoE models are Switch Transformers (Fedus et al., 2022). As a standard MoE setup we used 8 Experts, 1 of which is activated per token. All models have Attention head dimension of 64. Two auxiliary losses are used for the Router: a z-loss weighted at 0.001 (Zoph et al., 2022) and load balancing weighted at 0.01 (Fedus et al., 2022). All models for 3 have 24 layers and are trained for $16B$ tokens. Experiments for MoE are smaller for technical reasons. All Models in 1 are trained for 1B tokens. They have 8 transformer layers. Models in 2 have 12 blocks, trained for 2.5B tokens.