μnit Scaling:
Simple and Scalable FP8 LLM Training

ICML 2025

Saaketh Narayan[1], Abhay Gupta[2], Mansheej Paul[1], Davis Blalock[2]

[1]Work done while at Databricks Mosaic Research [2]Databricks Mosaic Research

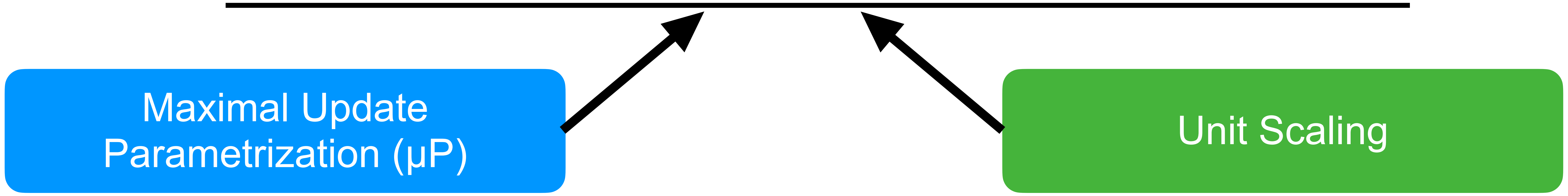
Low-Precision LLM Training

Method	Uses FP8	Hparam transfer	Number of Hparams	No dynamic scaling factors	Scales stably to large models	Training-Inference precision match	Efficient distributed training
BF16 mixed precision (SP)[1]	No	No	3	Yes	Yes	No	Yes
Maximal Update Parametrization (μP)[2]	No	Yes	6	Yes	Yes	No	Yes
Unit Scaling / u-μP[3,4]	Partially	Yes (u-μP)	7	Yes	Partially	Partially	Partially
Dynamically Scaled FP8 (SP), e.g. TE[5]	Yes	No	3	No	Partially	Yes	Yes
μnit Scaling (ours)	**Yes**	**Yes**	**3**	**Yes**	**Yes**	**Yes**	**Yes**

- Training LLMs is resource intensive, using FP8 promises significant efficiency gains
- Existing low-precision training schemes have various drawbacks
- Our method, **μnit Scaling (μS)**, combines full FP8 training with hparam transfer in a simple, straightforward, and scalable way

[1] Micikevicius, P., Narang, S., Alben, J., Diamos, G., Elsen, E., Garcia, D., Ginsburg, B., Houston, M., Kuchaiev, O., Venkatesh, G., and Wu, H. Mixed precision training. In International Conference on Learning Representations, 2018
[2] Yang, G., Hu, E. J., Babuschkin, I., Sidor, S., Liu, X., Farhi, D., Ryder, N., Pachocki, J., Chen, W., and Gao, J. Tuning large neural networks via zero-shot hyper-parameter transfer. Advances in Neural Information Processing Systems, 2021.
[3] Blake, C., Orr, D., and Luschi, C. Unit scaling: Out-of-the-box low-precision training. In International Conference on Machine Learning, pp. 2548–2576. PMLR, 2023.
[4] Blake, C., Eichenberg, C., Dean, J., Balles, L., Prince, L. Y., Deiseroth, B., Cruz-Salinas, A. F., Luschi, C., Weinbach, S., and Orr, D. u-μp: The unit-scaled maximal update parametrization. In WANT@ICML 2024, 2024
[5] NVIDIA. TransformerEngine, 2023.

The μS training scheme



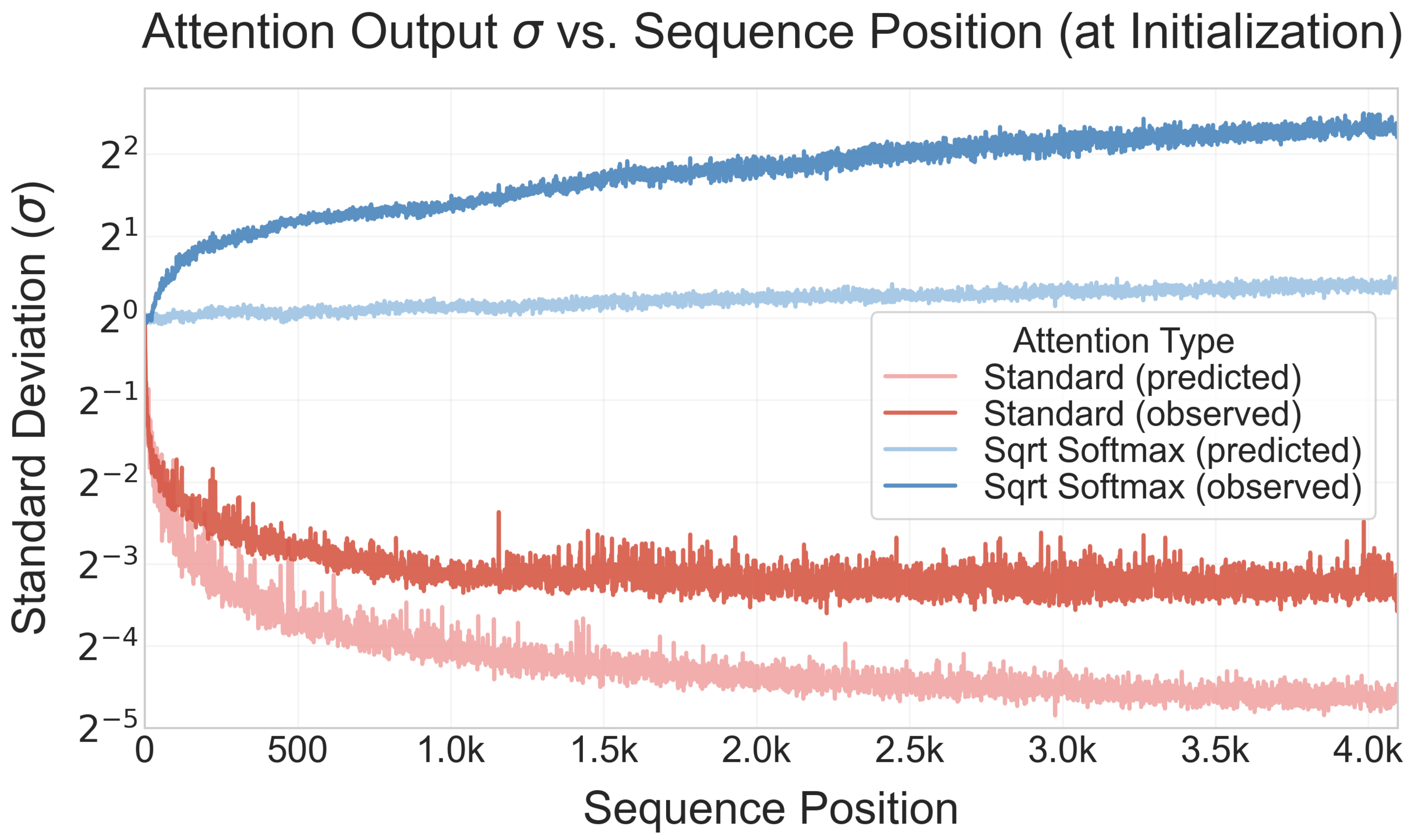
Modification	Description
Linear layer scaling factors	$\frac{1}{\sqrt{\text{fan\_in}}}$ static scaling factor applied in *both* forward and backward pass. The final LM head uses a multiplier of $\frac{1}{\text{fan\_in}}$ instead, in line with μP.
Res-Post-LayerNorm	LayerNorm is the last operation in each residual branch instead of the first.
"Fixed" residual modification	Use a fixed constant τ to make residuals variance-preserving, according to Eq. 11.
Unit variance initialization	All linear layer weights initialized with variance 1.
FP8 hidden layers	Use FP8E4M3 for weights and activations, FP8E5M2 for gradients. Before casting, clip BF16 values to FP8 dtype max. Keep embedding table and LM head in BF16.
Learning rate (η) scaling	Optimal η stays constant for input and output layers, but is scaled by $\frac{\sqrt{d_{\text{base}}}}{\sqrt{d_{\text{model}}}}$ for all hidden layers, when transferring from a base model with width d_{base}
Weight decay (λ) scaling	With fully decoupled weight decay, optimal λ stays constant for all layers with increasing width.

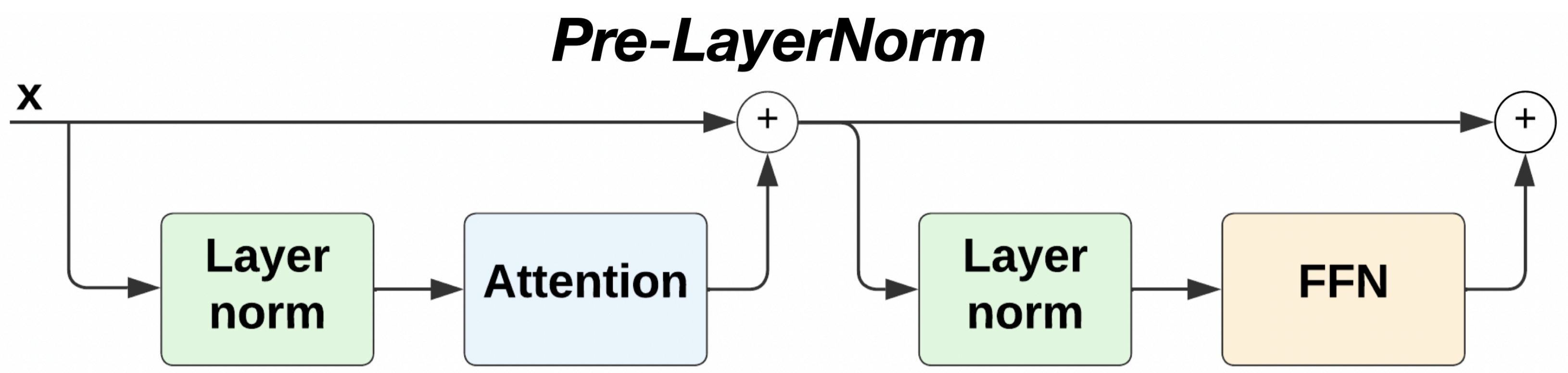
Poor numerics in Transformers: Self-attention

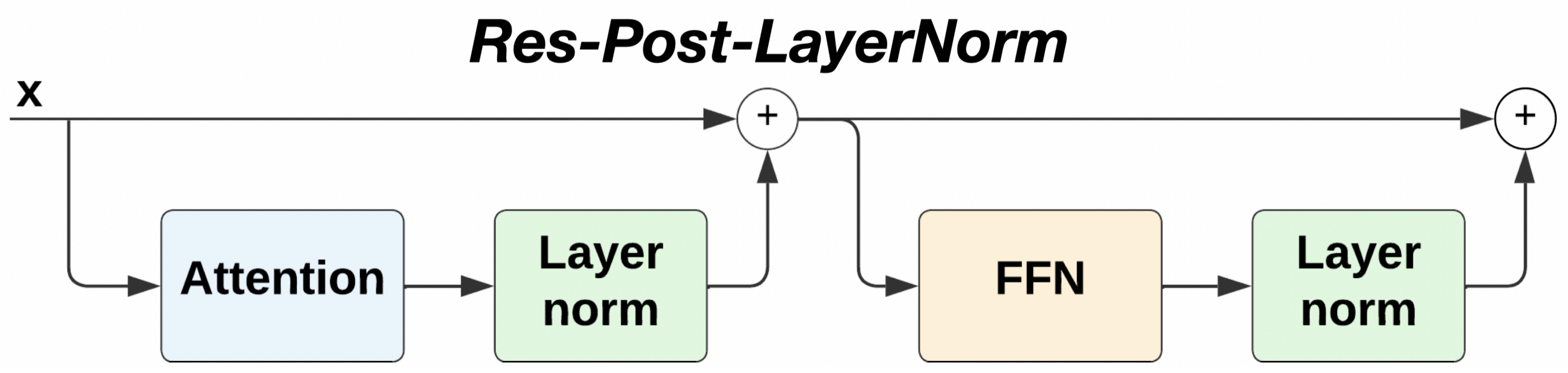
$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right)\mathbf{V}$$

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \sqrt{\text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)}\mathbf{V}$$

- Causal self-attention is not variance preserving, making low-precision training difficult
- Masking of attention logits matrix leads to output variance inversely proportional to a token's sequence position
- Simply taking square-root of logits also insufficient due to repeated / highly correlated value tokens in sequence data
- Proposed solution: Use Res-Post-LayerNorm[6] to normalize variance of attention outputs







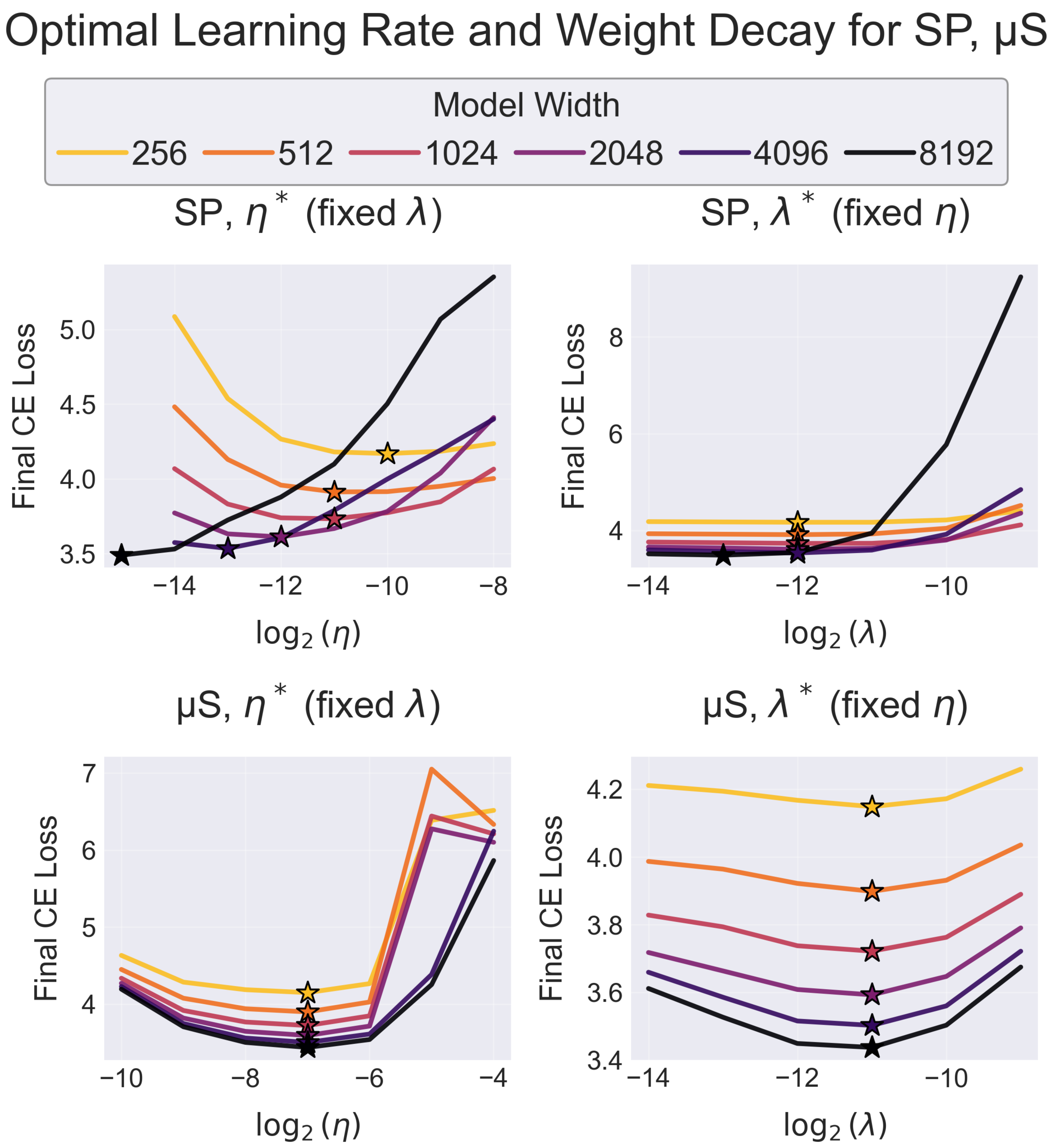
[6] Liu, Z., Hu, H., Lin, Y., Yao, Z., Xie, Z., Wei, Y., Ning, J., Cao, Y., Zhang, Z., Dong, L., et al. Swin transformer v2: Scaling up capacity and resolution. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 12009–12019, 2022.

Hyperparameter Transfer of both η and λ

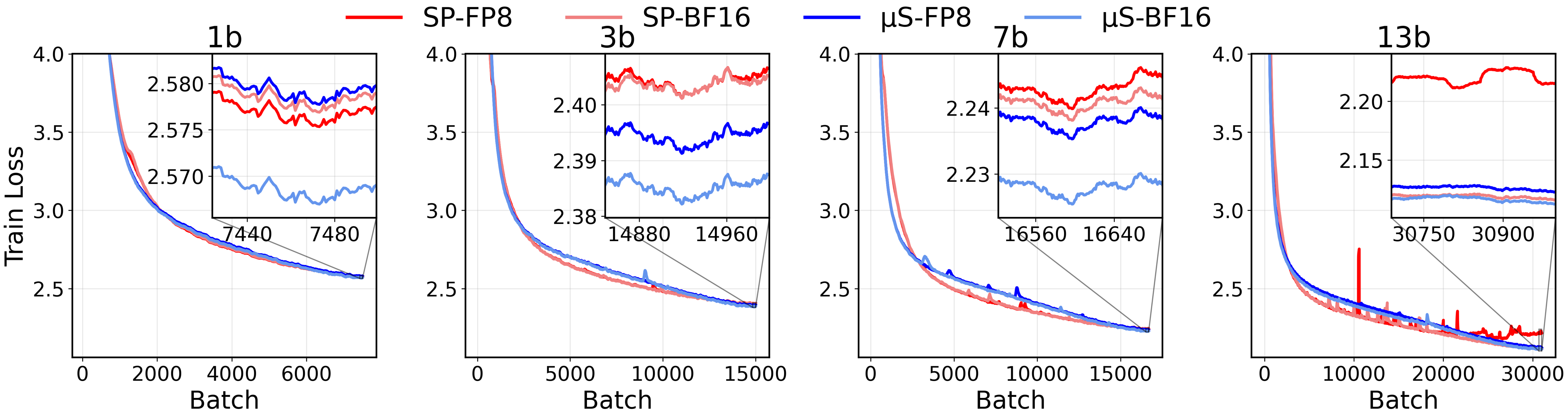
- Both learning rate (η) and weight decay (λ) are important for optimal LLM training
- μS demonstrates consistent hparam transfer of η and λ by combining Unit Scaling[3] with the Maximal Update Parametrization (μP)[2], as similarly shown in u-μP[4]
- μS requires tuning much fewer hparams than μP and u-μP
  - Hparam τ makes the residual connection variance-preserving:

$$\text{fixed}(\tau): x_{l+1} = \sqrt{1-\tau}\cdot x_l + \sqrt{\tau}\cdot f(x_l)$$

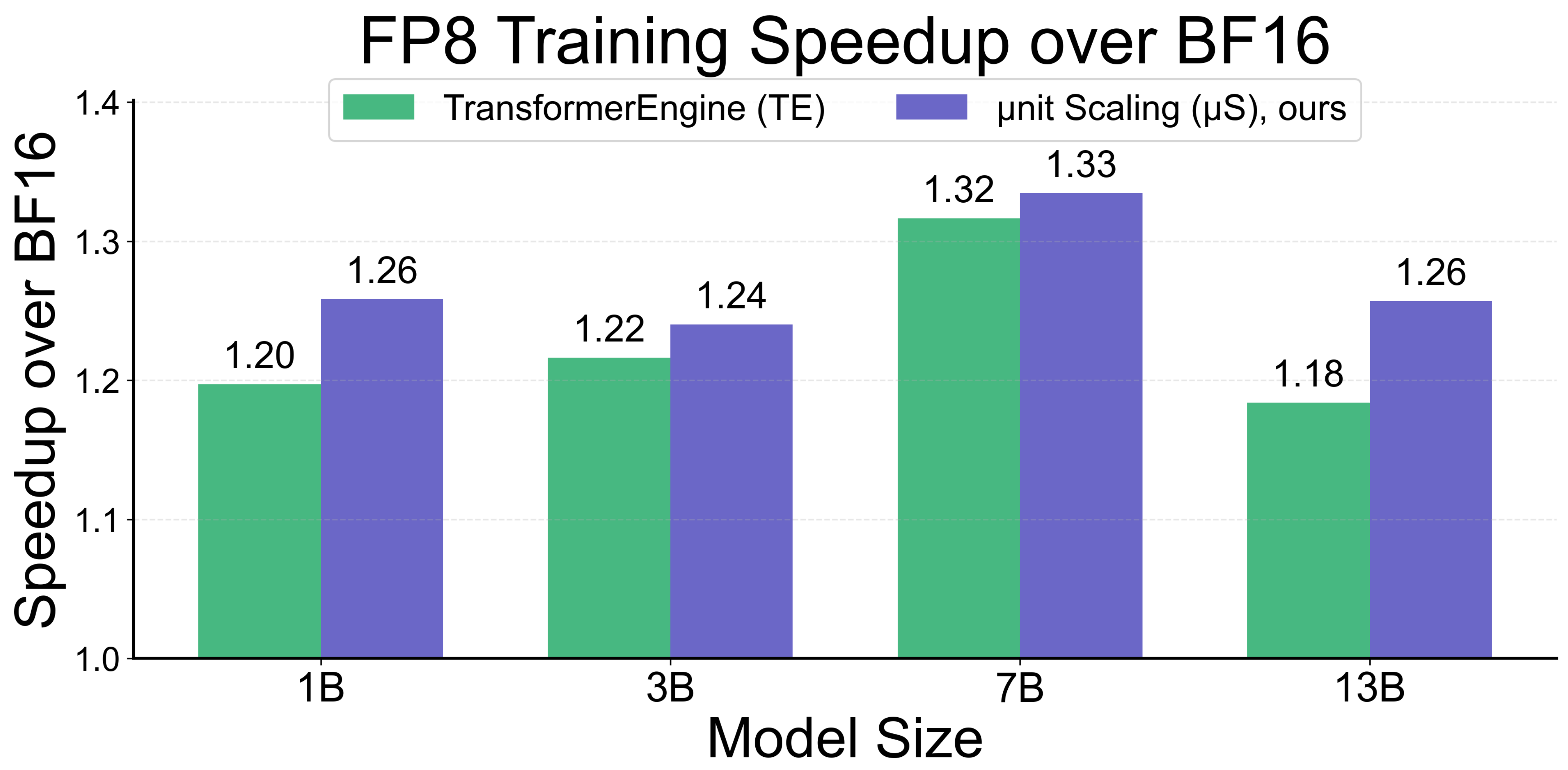
Scheme	# Hparams	Hparams
μS (ours)	3	η, λ, τ
SP	3	$\eta, \lambda, \sigma_{\text{init}}$
μP	6	$\eta, \lambda, \sigma_{\text{init}}, \alpha_{\text{res}}, \alpha_{\text{attn}}, \alpha_{\text{out}}$
u-μP	7	$\eta, \lambda, \alpha_{\text{ffn-act}}, \alpha_{\text{attn-softmax}}, \alpha_{\text{res}}, \alpha_{\text{res-attn-ratio}}, \alpha_{\text{loss-softmax}}$



Large-scale LLM training in FP8



- µS models successfully train in FP8 up to 13B scale
  - **All** transformer backbone matmuls done in FP8
- µS FP8 models converge similarly to BF16 counterparts
- SP 13B model in FP8 (with TransformerEngine) failed to converge
- µS provides state-of-the-art training efficiency.
  - Elimination of dynamic scaling overhead makes it faster than TE



µnit Scaling:
Simple and Scalable FP8 LLM Training

ICML 2025

Saaketh Narayan[1], Abhay Gupta[2], Mansheej Paul[1], Davis Blalock[2]

[1]Work done while at Databricks Mosaic Research [2]Databricks Mosaic Research