Effective Sharpness Aware Minimization Requires Layerwise Perturbation Scaling

Moritz Haas[1] **Jin Xu**[2] **Volkan Cevher**[3,4] **Leena Chennuru Vankadara**[4]
[1] *University of Tübingen and Tübingen AI Center*[*]
[2] *University of Oxford*[*]
[3] *LIONS, EPFL*[*]
[4] *AGI Foundations, Amazon*

Abstract

Sharpness Aware Minimization (SAM) enhances performance across various neural architectures and datasets. As models are continually scaled up to improve performance, a rigorous understanding of SAM's scaling behavior is paramount. To this end, we study the infinite-width limit of neural networks trained with SAM, using the Tensor Programs framework. Our findings reveal that the dynamics of standard SAM effectively reduce to applying SAM solely in the last layer in wide neural networks, even with optimal hyperparameters. In contrast, we identify a unique parameterization with layerwise perturbation scaling, which we call *maximal update and perturbation parameterization* (μP^2), that ensures all layers are both feature learning and effectively perturbed in the limit. Through experiments with MLPs, ResNets and Vision Transformers, we empirically demonstrate that μP^2 is the only parameterization to achieve hyperparameter transfer of the joint optimum of learning rate and perturbation radius across model scales. Moreover, we provide an intuitive condition to derive μP^2 for other perturbation rules like Adaptive SAM and SAM-ON, also ensuring balanced perturbation effects across all layers.

1. Introduction

Sharpness Aware Minimization (SAM) [15] and its variants [28, 35] improve generalization across a range of neural architectures and datasets [8, 25]. In the SAM formulation, we minimize a given loss L between our prediction and the data y as a function of the network's weights W, where an adversary simultaneously maximizes the same loss by perturbing the weights within a budget ρ.

A standard SAM update for an L-hidden layer multi layer perceptron (MLP) is given by

$$W_{t+1}^l = W_t^l - \eta_l \nabla_{W^l} L\left(f\left(\xi_t; W_t + \varepsilon_t\right), y_t\right), \quad \text{with} \quad \varepsilon_t^l = \rho \cdot \frac{\nabla_{W^l} L(f(\xi_t; W_t), y_t)}{\|\nabla_{W^l} L(f(\xi_t; W_t), y_t)\|}, \tag{SAM}$$

where t is the iteration count and ε_t^l denotes the perturbation in the l-th MLP layer with width $n \in \mathbb{N}$, and where we define an L-hidden layer MLP iteratively via

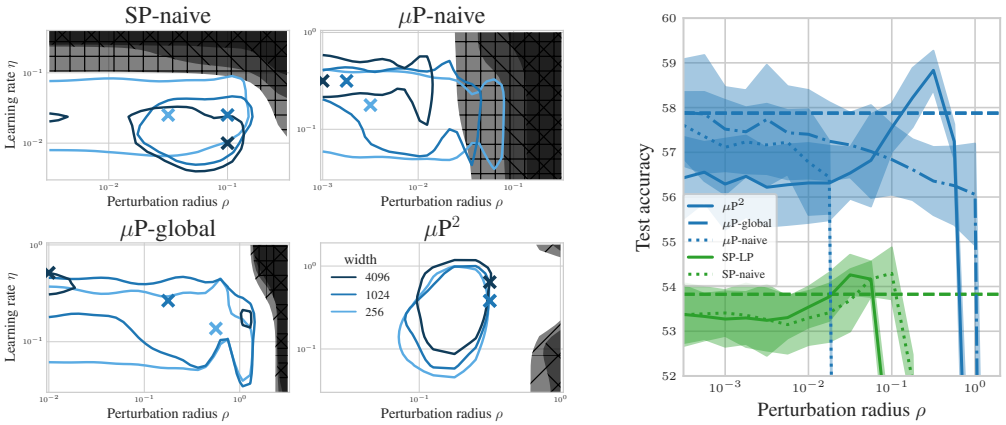
$$h^1(\xi) := W^1 \xi, \qquad x^l(\xi) := \phi(h^l(\xi)), \qquad h^{l+1}(\xi) := W^{l+1} x^l(\xi), \qquad f(\xi) := W^{L+1} x^L(\xi),$$

for inputs $\xi \in \mathbb{R}^{d_{\text{in}}}$ with trainable weight matrices $W^1 \in \mathbb{R}^{n \times d_{in}}$, $W^l \in \mathbb{R}^{n \times n}$ for $l \in [2, L]$, and $W^{L+1} \in \mathbb{R}^{d_{\text{out}} \times n}$. We call h^l preactivations, x^l activations, and $f(\xi)$ output function. Despite the inherent difficulty of non-convex, non-concave optimization, SAM is quite successful in practice.

[*] Moritz, Jin, and Volkan holds/held joint appointments at the University of Tübingen, University of Oxford, and EPFL respectively and Amazon. This work was done at Amazon. Correspondence to: mo.haas@uni-tuebingen.de

On the other hand, the steadily growing scale of foundation models has sparked considerable interest in scaling laws of model size and dataset size [26, 58]. To rigorously understand learning dynamics under width scaling, Yang and Hu [52] have recently provided general infinite-width theory for SGD, which has since been shown to be a good model for understanding the properties of large models [44]. Yang and Hu [52] show that standard parameterizations (SP), including He or LeCun initialization [19, 29] with a global learning rate, do not learn features in the infinite-width limit.

Instead, a different scaling of layerwise initialization variances and learning rates, termed *Maximal Update Parameterization* (μP), is necessary to achieve feature learning in wide networks. A crucial practical benefit of μP is the transferability of the optimal learning rate across model scales [54]. This can drastically reduce computational costs as it allows to tune hyperparameters on smaller representative models and then to train the large model only once. We provide a detailed account of related work and potential future work in Appendix I.



(*a*) (*b*)

Figure 1: *(a)* (**Only μP^2 transfers both η and ρ**): Test accuracy as a function of learning rate η and perturbation radius ρ of a 3-layer MLP trained with SAM on CIFAR10 for various widths and in different parameterizations (see subplot title), averaged over 3 independent runs.'$\times$' denotes the optimum. Blue contours (the darker, the wider) denote the region within 1% of the optimal test accuracy smoothened with a Gaussian filter. Grey regions (the lighter, the wider) denote the unstable regime below 30% test accuracy. 'naive' denotes no perturbation scaling, 'global' denotes global perturbation scaling $\rho = \Theta(n^{-1/2})$. *(b)* (**μP^2 achieves the best generalization performance**): Same as left but sliced at the optimal learning rate of each parameterization for width 4096. Dashed horizontal lines denote the base optimizer SGD in SP (green) and in μP (blue), respectively. Average and 2σ-CI from 16 independent runs. SP-LP denotes SP with layerwise perturbation scaling.

Contributions. In this paper, we adopt a scaling perspective to understand SAM's learning dynamics. Using the Tensor Programs framework [50, 52, 53], this work provides the first infinite-width theory for SAM with important practical consequences:

1. We show that training an MLP with the standard (SAM) update rule is equivalent to applying perturbations only in the last layer in the infinite-width limit even if the perturbation radius is properly tuned. This holds for any stable parameterization including SP and μP.

2. We postulate that just like learning rate transfers when features evolve non-trivially with width in every layer, transferability of the perturbation radius requires that every layer is *effectively perturbed* in the limit. We demonstrate that while the optimal learning rate transfers across widths under μP, the perturbation radius can shift significantly (Figure 1).
3. We show that this can be achieved by *layerwise scalings* of the perturbation radius. We characterize the class of parameterizations with layerwise scalings for which every layer of the network is effectively perturbed by SAM in the infinite-width limit.
4. We derive the unique *Maximal Update and Perturbation Parameterization* (μP^2) that achieves both feature learning and effective perturbations in all layers in the infinite-width limit. We empirically demonstrate that μP^2 is the only parameterization to achieve hyperparameter transfer in both learning rate η and perturbation radius ρ (Figure 1).
5. We provide a versatile scaling condition applicable to architectures such as ResNets and Vision Transformers (ViTs), and to various SAM variants like SAM-ON and Adaptive SAM (ASAM), and any SAM updates modeled in a Tensor Program.

2. Sharpness Aware Minimization in the infinite-width limit

2.1. SAM induces vanishing perturbations in wide neural networks

While our theory covers any stable parameterization including He and LeCun initializations, for concreteness and for the clarity of exposition, we first present our results for MLPs under μP:

$$\text{initialize} \quad W^1 \sim \mathcal{N}(0, 1/d_{in}),\ W^l \in \mathbb{R}^{n\times n} \sim \mathcal{N}(0, 1/n) \text{ for } l \in [2, L],\ W^{L+1} \sim \mathcal{N}(0, 1/n^2)$$
$$\text{with layerwise SGD learning rates} \quad \eta_1 = \eta n,\ \eta_l = \eta,\ \text{for } l \in [2, L],\ \eta_{L+1} = \eta n^{-1}.$$

By analyzing SAM's infinite-width behaviour, we show that the training dynamics under standard (SAM) become unstable under increasing width. All results are stated formally in Appendix III.

Proposition 1 (Instability of standard SAM parameterization in wide neural networks) *Under μP with the standard* (SAM) *update rule and default perturbation given in* (SAM)*, the output function becomes unbounded after the first update step in the infinite-width limit for any fixed, positive learning rate $\eta > 0$ and perturbation radius $\rho > 0$.*

To achieve stable optimization, it is necessary to introduce some width-dependent perturbation scaling ρn^{-d} for some suitable $d > 0$. Before we present our results on the width-scaling behavior of SAM under this scaling, we define the notion of *vanishing perturbations*.

Vanishing perturbations. The weight perturbation ε^l perturbs the l-th layer's activations as

$$x^l + \tilde{\delta}x^l = \phi((W^l + \varepsilon^l)(x^{l-1} + \tilde{\delta}x^{l-1})), \tag{1}$$

where $\tilde{\delta}x^l$ denotes the perturbation of the l-th layer's activations accumulated from the weight perturbations $\{\varepsilon^{l'}\}_{l'\in[l-1]}$ in all previous layers. We say a layer l has vanishing perturbations if $\tilde{\delta}x^l_t \to 0$ as the width approaches infinity. We omit time step t above for simplicity.

Informally, Theorem 2 below shows that for every choice of a decay parameter $d > 0$, either the training dynamics of SAM are unstable or all the hidden layers of the network have vanishing perturbations in the limit.

Theorem 2 (Global perturbation scaling is unstable or induces vanishing perturbations) *Fix $\rho > 0$ and $t \in \mathbb{N}$. Let $\mathring{f}_t$ denote the infinite-width limit of the output function after training an MLP of width n with the SAM update rule (SAM) with perturbation radius ρn^{-d} for t steps. If $d < 1/2$, then output perturbations blow up, and $\mathring{f}_t$ is unstable. If $d > 1/2$, then the perturbations in all layers vanish and $\mathring{f}_t$ corresponds to the limit after t steps of SGD. If $d = 1/2$, then only the last layer is effectively perturbed, all other layers have vanishing perturbations.*

Appendix VII.1 shows statistics of an MLP trained with (SAM) with global width-dependent scaling $\rho n^{-1/2}$ versus the same MLP trained with SAM where *only the last-layer weights are perturbed* and $\varepsilon^l = 0$ for all $l \in [L]$. As predicted by Theorem 2, both training algorithms produce equivalent training dynamics, already at moderate width, and last-layer perturbations are scaled correctly.

2.2. Effective perturbations using layerwise perturbation scaling

In this section, we show that correcting the (SAM) update rule to achieve *effective perturbations* in every single layer requires introducing additional hyperparameters — layerwise width-dependent scaling of the perturbation radius. This is similar in spirit to μP which corrects standard parameterization by introducing layerwise scaling of the learning rates. Under layerwise perturbation scaling, we show that there exists a unique parameterization, we call *maximal update and perturbation parameterization* (μP^2) that achieves both **feature learning and effective perturbations in all layers in the infinite-width limit**.

Formally, we extend the class of *abc*-parameterizations[1] [52] by including layerwise scaling of the perturbation radius. For clarity of exposition, we present our main results for MLPs. It is straightforward to extend our theory to any architecture that is representable as a NE⊗OR⊤program including ResNets and Transformers. Further theoretical considerations such as extensions to other ASAM variants and architectures can be found in Appendix V. For all of the results in this section, we assume that the used activation function is either `tanh` or σ`-gelu` for $\sigma > 0$ sufficiently small. For small enough $\sigma > 0$, σ`-gelu` (Definition 13) approximates ReLU arbitrarily well. The full formal result statements can be found in Appendix III. All proofs are provided in Appendix IV.

Definition 3 (*bcd*-parametrization) *A bcd-parametrization $\{b_l\}_{l\in[L+1]} \cup \{c_l\}_{l\in[L+1]} \cup \{d_l\}_{l\in[L+1]} \cup \{d\}$ defines the training of an MLP with SAM in the following way:*

(a) Initialize weights iid as $W^l_{ij} \sim \mathcal{N}(0, n^{-2b_l})$.
(b) Train the weights using the SAM update rule with layerwise learning rates,

$$W^l_{t+1} = W^l_t - \eta n^{-c_l} \nabla_{W^l} L\left(f\left(\xi_t; W_t + \varepsilon_t\right), y_t\right),$$

with the scaled perturbation ε_t via layerwise perturbation radii,

$$\varepsilon_t := \rho n^{-d} \frac{v_t}{\|v_t\|}, \quad \textit{with} \quad v_t = (v_t^1, \ldots, v_t^{L+1}), \quad v_t^l := n^{-d_l} \cdot \nabla_{W^l} L(f(\xi_t; W_t), y_t), \quad \text{(LP)}$$

W.l.o.g. we set $\|v_t\| = \Theta(1)$, which prevents nontrivial width-dependence from the denominator. This imposes the constraints: $d_1 \geq 1/2 - \min(b_{L+1}, c_{L+1})$, $d_l \geq 1 - \min(b_{L+1}, c_{L+1})$ for $l \in$

1. For each *abc*-parameterization $(a_l, b_l, c_l)_{l=1,\ldots,L+1}$, we consider the SGD-equivalent parameterization $(0, b_l + a_l, c_l + 2a_l)_{l=1,\ldots,L+1}$, which condenses all equations. In Appendix V.7, we derive a choice of weight multipliers for which global perturbation scaling induces effective perturbations in all layers.

$[2, L]$, *and* $d_{L+1} \geq 1/2$, *with at least one equality required to hold (see Appendix IV.1.3). The normalization* $v_t/\|v_t\|$ *removes one degree of freedom from* $\{d_l\}_{l\in[L+1]}$ *via the equivalence* $\{d'_l\}_{l\in[L+1]} \cong \{d_l\}_{l\in[L+1]}$ *iff there exists a* $C \in \mathbb{R}$ *such that* $d'_l = d_l + C$ *for all* $l \in [L+1]$.

Stability. We impose the same conditions on a *bcd*-parameterization to be *stable* as Yang and Hu [52] and additionally require that perturbations do not blow up. Altogether we call a *bcd*-parameterization *stable* (Definition 7) if the hidden activations have width-independent scaling $\Theta(1)$ at initialization and during training, and neither the updates nor the perturbations $\tilde{\delta}x^l$ of the activations or output logits $f_{\tilde{W}_t} - f_{W_t}$ blow up at any point in training.

Effective perturbations. Given stability, the choice of layerwise initialization variance and learning rate scalings $\{b_l\}_{l\in[L+1]} \cup \{c_l\}_{l\in[L+1]}$ is largely decoupled from the choice of layerwise perturbation scaling $\{d_l\}_{l\in[L+1]} \cup \{d\}$. To achieve feature learning in every layer, μP is the unique choice of $\{b_l\}_{l\in[L+1]} \cup \{c_l\}_{l\in[L+1]}$. What is left to do is to find the unique choice of $\{d_l\}_{l\in[L+1]} \cup \{d\}$ so that the perturbations in every layer have a non-vanishing and non-exploding effect on the output. Recalling (1), it is crucial to not only propagate non-vanishing perturbations $\tilde{\delta}x^{l-1}$ through the forward pass, but for the l-th layer weights to contribute a non-vanishing term $\varepsilon^l(x^{l-1} + \tilde{\delta}x^{l-1})$. If $\varepsilon^l(x^{l-1} + \tilde{\delta}x^{l-1}) = \Theta(1)$, we say that the l-th layer is *effectively perturbed*. Given μP with $\min(b_{L+1}, c_{L+1}) = 1$, the following theorem provides the perturbation scaling to reach μP^2.

Theorem 4 (Maximal Perturbation Parameterization (MPP)) *Consider any stable bcd-parametrization* $\{b_l\}_{l\in[L+1]} \cup \{c_l\}_{l\in[L+1]} \cup \{d_l\}_{l\in[L+1]} \cup \{d\}$ *with* $b_{L+1} \geq 1$. *Then up to the equivalence* $d'_l = d_l + C$, $C \in \mathbb{R}$, $\forall l \in [L+1]$, *the unique stable choice* $\{d_l\}_{l\in[L+1]} \cup \{d\}$ *that effectively perturbs all layers* $l \in [L+1]$ *is given by*

$$d = -1/2, \qquad d_l = \begin{cases} 1/2 - \min(b_{L+1}, c_{L+1}) & l = 1, \\ 3/2 - \min(b_{L+1}, c_{L+1}) & l \in [2, L], \\ 3/2 & l = L+1. \end{cases} \tag{2}$$

3. μP^2 achieves hyperparameter transfer and improved generalization

Figure 1 shows that only μP^2 yields hyperparameter transfer in (η, ρ) and improved generalization for MLPs trained on CIFAR10 [27]. Figure 2 shows that μP^2 also improves performance in ResNets and stabilizes SAM's training dynamics.

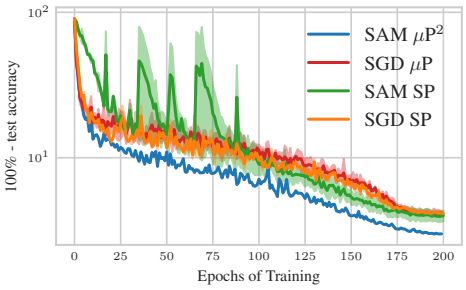


Figure 2: **(ResNets in μP^2)** Training a ResNet-18 with width multiplier 2 with SAM in μP^2 versus SP on CIFAR10.

An intuitive condition for generalizing μP^2 to other gradient-based perturbation rules is: weight perturbations of every layer should scale like their updates, $\varepsilon_t^l = \Theta(\delta W_t^l)$. In Appendix V.5, we apply this condition to SAM-ON [35] and ASAM [28] in ResNet-18 [20] trained on CIFAR10, and find that μP^2 improves the generalization performance of all considered SAM variants compared to SP. In μP^2, most SAM variants perform similarly well, suggesting that their differences in SP primarily stem from differing degrees to which the normalization layers are perturbed. Experimental details are disclosed in Appendix VI and supplemental experiments including ρ-transfer in ViTs on Imagenet1K can be found in Appendix VII.

References

[1] Maksym Andriushchenko and Nicolas Flammarion. Towards understanding sharpness-aware minimization. In *International Conference on Machine Learning*, pages 639–668. PMLR, 2022.

[2] Maksym Andriushchenko, Dara Bahri, Hossein Mobahi, and Nicolas Flammarion. Sharpness-aware minimization leads to low-rank features. *arXiv:2305.16292*, 2023.

[3] Maksym Andriushchenko, Francesco Croce, Maximilian Müller, Matthias Hein, and Nicolas Flammarion. A modern look at the relationship between sharpness and generalization. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 840–902. PMLR, 23–29 Jul 2023.

[4] Sanjeev Arora, Zhiyuan Li, and Abhishek Panigrahi. Understanding gradient descent on the edge of stability in deep learning. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 948–1024, 17–23 Jul 2022.

[5] Peter L. Bartlett, Philip M. Long, and Olivier Bousquet. The dynamics of sharpness-aware minimization: Bouncing across ravines and drifting towards wide minima. *Journal of Machine Learning Research*, 24(316):1–36, 2023.

[6] Tamay Besiroglu, Ege Erdil, Matthew Barnett, and Josh You. Chinchilla scaling: A replication attempt. *arXiv:2404.10102*, 2024.

[7] Blake Bordelon, Lorenzo Noci, Mufan Bill Li, Boris Hanin, and Cengiz Pehlevan. Depthwise hyperparameter transfer in residual networks: Dynamics and scaling limit. *arXiv:2309.16620*, 2023.

[8] Xiangning Chen, Cho-Jui Hsieh, and Boqing Gong. When vision transformers outperform resnets without pre-training or strong data augmentations. *arXiv:2106.01548*, 2021.

[9] Jeremy Cohen, Simran Kaur, Yuanzhi Li, J Zico Kolter, and Ameet Talwalkar. Gradient descent on neural networks typically occurs at the edge of stability. In *International Conference on Learning Representations*, 2020.

[10] Yan Dai, Kwangjun Ahn, and Suvrit Sra. The crucial role of normalization in sharpness-aware minimization. *Advances in Neural Information Processing Systems*, 36, 2024.

[11] Yann N Dauphin, Atish Agarwala, and Hossein Mobahi. Neglected hessian component explains mysteries in sharpness regularization. *arXiv:2401.10809*, 2024.

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE conference on computer vision and pattern recognition (ICCV)*, pages 248–255, 2009.

[13] Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1019–1028, 06–11 Aug 2017.

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[15] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2021.

[16] Boris Hanin and David Rolnick. How to start training: The effect of initialization and architecture. *Advances in neural information processing systems*, 31, 2018.

[17] Soufiane Hayou and Greg Yang. Width and depth limits commute in residual networks. In *International Conference on Machine Learning*, pages 12700–12723. PMLR, 2023.

[18] Soufiane Hayou, Eugenio Clerico, Bobby He, George Deligiannidis, Arnaud Doucet, and Judith Rousseau. Stable resnet. In *International Conference on Artificial Intelligence and Statistics*, pages 1324–1332. PMLR, 2021.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *IEEE international conference on computer vision (ICCV)*, pages 1026–1034, 2015.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[21] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Comput.*, 9(1):1–42, jan 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.1.1.

[22] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

[23] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural Tangent Kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8571–8580, 2018.

[24] Yiding Jiang*, Behnam Neyshabur*, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2020.

[25] Jean Kaddour, Linqing Liu, Ricardo Silva, and Matt J Kusner. When do flat minima optimizers work? *Advances in Neural Information Processing Systems*, 35:16577–16595, 2022.

[26] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv:2001.08361*, 2020.

[27] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[28] Jungmin Kwon, Jeongseop Kim, Hyunseo Park, and In Kwon Choi. Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. In *International Conference on Machine Learning*, pages 5905–5914. PMLR, 2021.

[29] Yann LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–50. Springer, 2002.

[30] Mufan Li, Mihai Nica, and Dan Roy. The future is log-gaussian: Resnets and their infinite-depth-and-width limit at initialization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 7852–7864, 2021.

[31] Yong Liu, Siqi Mai, Xiangning Chen, Cho-Jui Hsieh, and Yang You. Towards efficient and scalable sharpness-aware minimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12360–12370, 2022.

[32] Philip M. Long and Peter L. Bartlett. Sharpness-aware minimization and the edge of stability. *arXiv:2309.12488*, 2023.

[33] Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33): E7665–E7671, 2018.

[34] Enea Monzio Compagnoni, Luca Biggio, Antonio Orvieto, Frank Norbert Proske, Hans Kersting, and Aurelien Lucchi. An SDE for modeling SAM: Theory and insights. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 25209–25253, 23–29 Jul 2023.

[35] Maximilian Müller, Tiffany Vlaar, David Rolnick, and Matthias Hein. Normalization layers are all that sharpness-aware minimization needs. *Advances in Neural Information Processing Systems*, 36, 2024.

[36] Radford M. Neal. *Priors for Infinite Networks*, pages 29–53. Springer New York, 1996.

[37] Lorenzo Noci, Sotiris Anagnostidis, Luca Biggio, Antonio Orvieto, Sidak Pal Singh, and Aurelien Lucchi. Signal propagation in transformers: Theoretical perspectives and the role of rank collapse. *Advances in Neural Information Processing Systems*, 35:27198–27211, 2022.

[38] Lorenzo Noci, Chuning Li, Mufan Li, Bobby He, Thomas Hofmann, Chris J Maddison, and Dan Roy. The shaped transformer: Attention models in the infinite depth-and-width limit. *Advances in Neural Information Processing Systems*, 36, 2024.

[39] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style,

high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[40] Ben Poole, Subhaneil Lahiri, Maithra Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. *Advances in neural information processing systems*, 29, 2016.

[41] David Samuel. (adaptive) sam optimizer (pytorch). https://github.com/davda54/sam, 2022.

[42] Samuel S Schoenholz, Justin Gilmer, Surya Ganguli, and Jascha Sohl-Dickstein. Deep information propagation. *arXiv:1611.01232*, 2016.

[43] Sungbin Shin, Dongyeop Lee, Maksym Andriushchenko, and Namhoon Lee. The effects of overparameterization on sharpness-aware minimization: An empirical and theoretical analysis. *arXiv:2311.17539*, 2023.

[44] Nikhil Vyas, Alexander Atanasov, Blake Bordelon, Depen Morwani, Sabarish Sainathan, and Cengiz Pehlevan. Feature-learning networks are consistent across widths at realistic scales. *Advances in Neural Information Processing Systems*, 36, 2024.

[45] Kaiyue Wen, Tengyu Ma, and Zhiyuan Li. How sharpness-aware minimization minimizes sharpness? In *The Eleventh International Conference on Learning Representations*, 2023.

[46] Kaiyue Wen, Zhiyuan Li, and Tengyu Ma. Sharpness minimization algorithms do not only minimize sharpness to achieve better generalization. *Advances in Neural Information Processing Systems*, 36, 2024.

[47] Jonathan Wenger, Felix Dangel, and Agustinus Kristiadi. On the disconnect between theory and practice of overparametrized neural networks. *arXiv:2310.00137*, 2023.

[48] Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Zhicheng Yan, Masayoshi Tomizuka, Joseph Gonzalez, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision. *arXiv:2006.03677*, 2020.

[49] Lechao Xiao, Jeffrey Pennington, and Samuel Schoenholz. Disentangling trainability and generalization in deep neural networks. In *International Conference on Machine Learning*, pages 10462–10472. PMLR, 2020.

[50] Greg Yang. Wide feedforward or recurrent neural networks of any architecture are gaussian processes. *Advances in Neural Information Processing Systems*, 32, 2019.

[51] Greg Yang. Tensor programs iii: Neural matrix laws. *arXiv:2009.10685*, 2021.

[52] Greg Yang and Edward J. Hu. Tensor programs iv: Feature learning in infinite-width neural networks. In *International Conference on Machine Learning (ICML)*, 2021.

[53] Greg Yang and Etai Littwin. Tensor programs ivb: Adaptive optimization in the infinite-width limit. *arXiv:2308.01814*, 2023.

[54] Greg Yang, Edward J Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tensor programs v: Tuning large neural networks via zero-shot hyperparameter transfer. *arXiv:2203.03466*, 2022.

[55] Greg Yang, James B. Simon, and Jeremy Bernstein. A spectral condition for feature learning. *arXiv preprint arXiv:2310.17813*, 2023.

[56] Greg Yang, Dingli Yu, Chen Zhu, and Soufiane Hayou. Tensor programs vi: Feature learning in infinite-depth neural networks. *arXiv preprint arXiv:2310.02244*, 2023.

[57] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes. In *International Conference on Learning Representations*, 2020.

[58] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12104–12113, 2022.

Appendices

Appendix Contents.

I. Detailed related work

Signal propagation. Our work can be seen as scaling theory with the goal of preventing both vanishing and exploding signals in forward and backward passes, where the analysis of SAM requires considering stability of perturbations in each layer as well. In this sense, we build on a rich literature, often restricted to an analysis at initialization [16, 40, 42, 49]. For scaling neural networks to infinite depth, residual connections have been found to be beneficial for stabilizing signal propagation while retaining expressivity. The simple $\frac{1}{\sqrt{L}}$-scaling allows depth-scaling in ResNets and unlocks hyperparameter transfer [7, 18, 30, 56]. Noci et al. [37, 38] provide infinite width and depth analyses for Transformers with the goal of preventing rank collapse and attaining a limit that has behaviour consistent with that of moderately large networks.

Tensor Programs. After kernel-based approaches to understand infinite-width limits of neural networks [23, 36] and applications of mean-field theory [33], the Tensor Program series [50, 52–54, 56] marks the first important break through in the theory of large neural networks. The framework covers many modern deep learning architectures, optimization algorithms and arbitrary abc-parameterizations, where each abc-parameterization is essentially defined by a layerwise scaling of initialization variance and learning rate as a function of network width. Yang and Hu [52] propose the *maximal update parameterization* (μP) and show that it is the unique stable parameterization that achieves feature learning in all layers in the limit of infinite width. In this framework, training neural networks with a global learning rate $\eta > 0$ for all layers and with He or LeCun initialization falls under the category of so called *standard parameterization* (SP). The neural tangent parameterization (NTP), studied in the neural tangent kernel literature, differs but does not achieve feature learning in any layer, and is therefore less useful to describe the behaviour of finite width networks than μP [44, 47]. Yang and Littwin [53] characterize stable learning with adaptive optimizers at infinite width into a feature learning versus a (nonlinear) operator regime. SAM is not covered by the update rule definition in Yang and Littwin [53] since the nested application of the gradient w.r.t. the weights is not a coordinatewise optimizer anymore. While recent works have considered joint limits of infinite width and depth [17, 56], the data distribution has not been taken into account in Tensor Program literature. The study of scaling laws of jointly scaling model size, data set size and training time has predominantly been empirical [6, 22, 26, 58]. Developing theory to inform Pareto optimal trade offs in a principled manner constitutes an important direction for future work.

Sharpness Aware Minimization. Sharpness aware minimization (SAM) [15] has shown to be extremely effective and robust in improving generalization performance across a wide range of architectures and settings [8, 25]. SAM was motivated as an inductive bias towards flatter minima and it has been understood to have an gradient-norm adaptive edge of stability at which it drifts towards minima with smaller spectral norm of the Hessian [5, 32]. However a full understanding of why SAM works so well remains elusive. While correlations between flatness and generalization have been observed in some settings [21, 24], other studies have questioned the usefulness of sharpness as a measure for generalization, especially for modern architectures [3, 13, 46]. Applying SAM on only the normalization layers often even improves generalization in vision tasks depsite increasing sharpness [35]. Adaptive SAM (ASAM) [28] is a variant of SAM derived from a sharpness definition that is invariant to weight rescalings with respect to a chosen normalization operator that leave the output function invariant. The results in Müller et al. [35] suggest that two of the most promising normalization operators are elementwise normalization $T^l_w(x) = |W^l| \odot x$ and layerwise

normalization $T_w^l(x) = \|W^l\|_F \cdot x$. We state the resulting update rules and a scaling analysis in Appendix V.5. A variant of SAM that is often studied theoretically because of its simplicity does not normalize the gradient of the perturbation. Our theory covers this variant too (Appendix V.4), but Dai et al. [10] argue that normalizing the gradients for the perturbation is crucial. Monzio Compagnoni et al. [34] find that unnormalized SAM gets stuck around saddles while SAM slowly escapes through additional Hessian-induced noise. This suggests that the additional effort of analysing the original SAM update rule with gradient normalization is necessary for practically useful theory. Dauphin et al. [11] draw connections between SAM and other second order optimizers like gradient penalties and weight noise. They show that SAM is able to effectively use second order information implicitly using ReLU, whereas the other two methods close the gap to SAM when using GeLU since they require the localized second order information that GeLU provides in contrast to ReLU. Wen et al. [45] show that worst-case, ascent and average case sharpness are biased towards minimizing the maximal eigenvalue, minimal non-zero eigenvalue and trace of the Hessian, respectively. With an architecture-agnostic analysis, they show that 1-SAM minimizes the trace of Hessian like average-case sharpness, for small enough η and ρ. Similarly, the theoretical results by Andriushchenko and Flammarion [1] rely on the assumption that learning rate η and perturbation radius ρ are chosen sufficiently close to 0. Arguably, the empirically optimal choice of η and ρ lies outside of this gradient flow-like regime and has qualitatively different properties (see e.g. edge of stability literature [4, 9]).

Scaling theory for SAM. Shin et al. [43] suggest that the generalisation improvement by SAM continues to increase with growing overparametrization. This corroborates empirical observations that performance monotonically improves with scale, and understanding the infinite-width limit is not only of theoretical interest but entails immediate practical benefits.

Liu et al. [31] introduce Look-LayerSAM with layerwise perturbation scaling for preserving good performance under large batch training for enhanced training parallelization. They use LAMB [57] for layerwise learning rate scaling for large batch training. The update scaling strategy in these kinds of algorithms follows

$$W_{t+1}^l = W_t^l - \eta_t \phi(\|W_t^l\|_F) \frac{\nabla_{W^l} L}{\|\nabla_{W^l} L\|_F},$$

with some $\phi : \mathbb{R}_+ \to \mathbb{R}_+$ and where $\nabla_{W^l} L$ may be replaced by ADAM's $\frac{m_t}{\sqrt{v_t}+\varepsilon}$. In practice, often simple functions like $\phi(x) = \max(c, \min(x, C))$ or $\phi(x) = x$ are used. The idea is to ensure that the update has the same order of magnitude as the weights. Look-LayerSAM follows an analogous approach for layerwise perturbation scaling. A derivation of μP for LAMB could also yield feature learning in all layers in the infinite-width limit as well as hyperparameter transfer. It certainly requires layerwise learning rate scaling. In the case $\phi(x) = x$, following a heuristic scaling derivation as in Appendix V.5 leads to layerwise learning rate scalings $\eta_1 = \eta_{L+1} = \Theta(1)$ and $\eta_l = \Theta(n^{-1/2})$ for hidden layers $l \in [2, L]$. With a bounded function like $\phi(x) = \max(c, \min(x, C))$, the scalings become $\eta_1 = \Theta(n^{1/2})$, $\eta_{L+1} = \Theta(n^{-1/2})$ and $\eta_l = \Theta(1)$ for hidden layers $l \in [2, L]$. We leave a closer investigation of feature learning and hyperparameter transfer with LAMB and Look-LayerSAM in SP and μP to future work.

Future work. This study may serve as an inspiration of how scaling theory can be used to understand and improve training procedures in minimax optimization and beyond. To reach a fully practical theory of deep learning, it will be necessary to take data distributions and training dynamics into account in more detail than it is possible with current Tensor Program theory. For example, existing Tensor Program theory does not make statements about generalization. We also observe that ResNets

in SP can sometimes display HP transfer in η and ρ after training to convergence (Appendix VII.3.2). This contradicts the infinite-width theory from Yang and Hu [52] which predicts output blowup under large learning rates, and it shows that the exact conditions which enable hyperparameter transfer in practice are not fully understood. We plan to address some of these questions in upcoming work.

II. Definitions

In this section, we collect all definitions that do not appear in the main text. With minor modifications, we adopt all definitions from Yang and Hu [52]. If not stated otherwise, limits are taken with respect to width $n \to \infty$.

Definition 5 (Big-O Notation) *Given a sequence of scalar random variables $c = \{c_n \in \mathbb{R}\}_{n=1}^{\infty}$, we write $c = \Theta(n^{-a})$ if there exist constants A, B such that for almost every instantiation of $c = \{c_n \in \mathbb{R}\}_{n=1}^{\infty}$, for n large enough, $An^{-a} \leq |c_n| \leq Bn^{-a}$. Given a sequence of random vectors $x = \{x_n \in \mathbb{R}^n\}_{n=1}^{\infty}$, we say x has coordinates of size $\Theta(n^{-a})$ and write $x = \Theta(n^{-a})$ to mean the scalar random variable sequence $\left\{\sqrt{\|x_n\|^2/n}\right\}_n$ is $\Theta(n^{-a})$. Similarly for the notations $O(n^{-a}), \Omega(n^{-a})$. We write $x_n = o(n^{-a})$ if $n^a \cdot \sqrt{\|x_n\|^2/n} \to 0$ almost surely.*

Definition 6 (Training routine) *A training routine is a combination of base learning rate $\eta \geq 0$, perturbation radius $\rho \geq 0$, training sequence $\{(\xi_t, y_t)\}_{t \in \mathbb{N}}$ and a continuously differentiable loss function $L(f(\xi), y)$ using the SAM update rule with layerwise perturbation scaling* (LP)*.*

In addition to the stability conditions from the corresponding SGD result, we demand that the activation perturbations do not blow up. Otherwise the perturbations would strictly dominate both the initialization and the updates which makes the perturbation too strong and is avoided in practice.

Definition 7 (Stability) *We say a bcd-parametrization of an L-hidden layer MLP is stable if*

1. *For every nonzero input $\xi \in \mathbb{R}^{d_{in}} \backslash \{0\}$,*
$$h_0^l, x_0^l = \Theta_\xi(1),\ \forall l \in [L], \quad \text{and} \quad \mathbb{E} f_0(\xi)^2 = O_\xi(1),$$
where the expectation is taken over the random initialization.
2. *For any training routine, any time $t \in \mathbb{N}$, $l \in [L]$, $\xi \in \mathbb{R}^{d_{in}}$, we have*
$$h_t^l(\xi) - h_0^l(\xi), x_t^l(\xi) - x_0^l(\xi) = O_*(1), \quad \text{and} \quad f_t(\xi) = O_*(1),$$
where the hidden constant in O_* can depend on the training routine, t, ξ, l and the initial function f_0.
3. *For any training routine, any time $t \in \mathbb{N}_0$, $l \in [L]$, $\xi \in \mathbb{R}^{d_{in}}$, for the perturbed (pre-)activation $\tilde{h}_t^l := h^l(\tilde{W}_t), \tilde{x}_t^l := x^l(\tilde{W}_t)$ and output function $\tilde{f}_t(\tilde{W}_t)$ we have*
$$\tilde{h}_t^l(\xi) - h_t^l(\xi), \tilde{x}_t^l(\xi) - x_t^l(\xi) = O_*(1), \quad \text{and} \quad \tilde{f}_t(\xi) = O_*(1),$$
where the hidden constant in O_* can depend on the training routine, t, ξ, l and the initial function f_0.

Definition 8 (Nontriviality) *We say a bcd-parametrization is trivial if for every training routine, $f_t(\xi) - f_0(\xi) \to 0$ almost surely for $n \to \infty$, for every time $t > 0$ and input $\xi \in \mathbb{R}^{d_{in}}$. Otherwise the bcd-parametrization is nontrivial.*

Definition 9 (Feature learning) *We say a bcd-parametrization admits feature learning in the l-th layer if there exists a training routine, a time $t > 0$ and input ξ such that $x_t^l(\xi) - x_0^l(\xi) = \Omega_*(1)$, where the constant may depend on the training routine, the time t, the input ξ and the initial function f_0 but not on the width n.*

Definition 10 (Vanishing perturbations) *Let $l \in [L]$. We say that a stable bcd-parametrization has vanishing perturbations in the l-th layer if for any training routine, $t \in \mathbb{N}_0$ and $\xi \in \mathbb{R}^{d_{in}}$, it holds that $\tilde{x}_t^l - x_t^l = o(1)$, and it has vanishing perturbations in the output if for any training routine, $t \in \mathbb{N}_0$ and $\xi \in \mathbb{R}^{d_{in}}$ it holds that $\tilde{\delta} f_t(\xi) := f_{\tilde{W}_t}(\xi) - f_{W_t}(\xi) = o(1)$.*

Definition 11 (Perturbation nontriviality) *Let $l \in [L]$. We say that a stable bcd-parametrization is perturbation nontrivial with respect to the l-th layer if and only if it does not have vanishing perturbations in the l-th layer. A stable bcd-parametrization is perturbation nontrivial with respect to the output if it does not have vanishing perturbations in the output.*

Definition 12 (Effective perturbations) *Let $l \in [L+1]$. We say that a stable bcd-parametrization effectively perturbs the l-th layer if there exists a training routine, $t \in \mathbb{N}$ and $\xi \in \mathbb{R}^{d_{in}}$ such that $\tilde{\delta} W_t^l \tilde{x}_t^{l-1}(\xi) = \Theta(1)$ where $\tilde{\delta} W_t^l$ is defined in (LP) and $\tilde{x}_t^0 = x_t^0 = \xi_t$.*

Definition 13 (σ-gelu) *Define σ-gelu to be the function $x \mapsto \frac{x}{2}\left(1 + \operatorname{erf}\left(\sigma^{-1}x\right)\right) + \sigma\frac{e^{-\sigma^{-2}x^2}}{2\sqrt{\pi}}$.*

In order to apply the Tensor Program Master Theorem, all Nonlin and Moment operations in the NE⊗OR⊤ program, which do not only contain parameters as inputs, are required to be pseudo-Lipschitz in all of their arguments. For training with SGD, this is fulfilled as soon as ϕ' is pseudo-Lipschitz. Both `tanh` as well as σ`-gelu` fulfill this assumption.

Definition 14 (Pseudo-Lipschitz) *A function $f : \mathbb{R}^k \to \mathbb{R}$ is called pseudo-Lipschitz of degree d if there exists a $C > 0$ such that $|f(x) - f(y)| \leq C\|x - y\|(1 + \sum_{i=1}^k |x_i|^d + |y_i|^d)$. We say f is pseudo-Lipschitz if it is so for any degree d.*

III. Extensive main results

Using the formal definitions from Appendix II, here we provide the full formal statements of all of our main theoretical results together with further details and implications. The proof of all statements is provided in Appendix IV. Since SAM evaluates the gradients on perturbed weights, it is not covered by the update rule definition in Yang and Littwin [53] and an infinite-width analysis requires explicitly deriving the corresponding NE⊗OR⊤ program, scalings and infinite-width limits.

Recall that our definition of *bcd*-parameterizations extends *abc*-parameterizations by setting the maximal perturbation scaling to n^{-d} and allowing relative downweighting n^{-d_l} of the global scaling

in each layer l. The perturbation scaling does not affect the choice of layerwise initialization variance scalings b_l and the layerwise learning rate scalings c_l. Common bc-parametrizations for SGD are summarized in Table III.1. SAM with SGD as a base optimizer requires the same scalings. Similarly, SAM with ADAM as a base optimizer requires the same scalings as ADAM [54, Table 3]. Recall that, for convenience, we require width-independent denominator scaling $\|v_t\| = \Theta(1)$ of the scaled gradient for the perturbation (LP), which imposes the constraints

$$d_1 \geq 1/2 - \min(b_{L+1}, c_{L+1}), \quad d_l \geq 1 - \min(b_{L+1}, c_{L+1}) \text{ for } l \in [2, L], \quad d_{L+1} \geq 1/2. \tag{III.1}$$

All (pre-)activation and function outputs can be thought of as outputs given a fixed input $\xi \in \mathbb{R}^{d_{\text{in}}} \setminus \{0\}$ with $d_{in} \in \mathbb{N}$ fixed, e.g. $f_t := f_{W_t} := f_{W_t}(\xi)$. For the perturbed weights we write $\tilde{W}_t := W_t + \tilde{\delta} W_t$, with $\tilde{\delta} W_t$ defined in (LP) as ε_t^l. Here we write weight perturbations as $\tilde{\delta} W_t^l$ instead of ε_t^l to show the resemblance to weight updates δW_t^l. Perturbed activations and function outputs at time t are written as $\tilde{x}_t^l(\xi) = x^l_{\tilde{W}_t}(\xi)$ and $\tilde{f}_t(\xi) = f_{\tilde{W}_t^l}(\xi)$. Recall that for all of the results in this section we make the following smoothness assumption on the activation function.

Assumption 15 (Smooth activation function) *The used activation function is either* `tanh` *or* σ`-gelu` *for $\sigma > 0$ sufficiently small.*

For stating the conditions that characterize the class of stable bcd-parameterizations, we define the *maximal feature update scale* of a bcd-parameterization

$$r := \min(b_{L+1}, c_{L+1}, d + d_{L+1}) + \min_{l=1}^{L}(c_l - \mathbb{I}(l \neq 1)). \tag{III.2}$$

as well as the *maximal feature perturbation scale* of a bcd-parameterization

$$\tilde{r} := \min(b_{L+1}, c_{L+1}) + d + \min_{l=1}^{L}(d_l - \mathbb{I}(l \neq 1)). \tag{III.3}$$

As for SGD [52], the maximal feature update scaling r denotes the scaling of the last-layer activation updates δx^l. Ideally these updates should be non-exploding and non-vanishing. In that case, we call a stable parameterization *feature learning*, characterized by $r = 0$.

Similarly, $\tilde{r}$ describes how much the last hidden-layer activations x^L are perturbed as a function of width. Hidden-layer activation perturbations do not explode with width if and only if $\tilde{r} \geq 0$. Additionally, the output perturbations not to blow up if and only if $d + d_l \geq 1$ and $b_{L+1} + \tilde{r} \geq 1$. In particular, this implies that any stable bc-parameterization together with naive perturbation scaling $d_l = d = 0$ for all $l \in [L + 1]$ is *unstable due to blowup in the last layer*.

Stability requires the constraints (a-c) from SGD and additional perturbation stability constraints (d-e) that include the layerwise perturbation scales $\{d_l\}_{l=1,\dots,L+1}$.

Theorem 16 (Stability characterization) *A bcd-parametrization is stable if and only if all of the following are true:*

(a) (Stability at initialization, $h_0^l, x_0^l = \Theta(1)$ for all l, $f_0 = O(1)$)
$b_1 = 0$, $b_l = 1/2$ *for* $l \in [2, L]$ *and* $b_{L+1} \geq 1/2$.

(b) (Features do not blow up during training, i.e. $\Delta x_t^l = O(1)$ for all l)
$r \geq 0$.
(c) (Output function does not blow up during training, i.e. $\Delta W_t^{L+1} x_t^L, W_0^{L+1} \Delta x_t^L = O(1)$)
$c_{L+1} \geq 1$ *and* $b_{L+1} + r \geq 1$.
(d) (Feature perturbations do not blow up, i.e. $\tilde{\delta} x_t^l = O(1)$ for all l)
$\tilde{r} \geq 0$.
(e) (Output function perturbations do not blow up during training, i.e. $\tilde{\delta} W_t^{L+1} \tilde{x}_t^L, W_t^{L+1} \tilde{\delta} x_t^L = O(1)$)
$d + d_{L+1} \geq 1$ *and* $b_{L+1} + \tilde{r} \geq 1$.

The nontriviality and feature learning characterizations from SGD remain unaltered. This is because in the definition of r, it holds that $d + d_{L+1} \geq 1$ (from perturbation stability), and $\min(b_{L+1}, c_{L+1}) \leq 1$ already had to hold for nontriviality in SGD, so that stable perturbation scaling does not affect r.

Theorem 17 (Nontriviality characterization) *A stable bcd-parametrization is nontrivial if and only if $c_{L+1} = 1$ or $\min(b_{L+1}, c_{L+1}) + r = 1$.*

As for nontriviality, the conditions under which a stable, nontrivial parameterization is feature learning in the infinite-width limit are decoupled from the choice of perturbation scalings $\{d_l\}_{l \in [L+1]} \cup \{d\}$. Hence the conditions are the same as for SGD. Below we provide a slightly refined result in terms of the maximal feature update scale r_{l_0} of a bcd-parameterization up to layer l_0 (as provided in the Appendix of Yang and Hu [52]).

Theorem 18 (Feature learning characterization) *For any $l_0 \in [L]$, the following statements are equivalent:*

(a) A stable, nontrivial bcd-parametrization admits feature learning in layer l_0.
(b) A stable, nontrivial bcd-parametrization admits feature learning in layer l for all $l \geq l_0$.
(c) $r_{l_0} := \min(b_{L+1}, c_{L+1}, d + d_{L+1}) + \min_{m=1}^{l_0}(c_m - \mathbb{I}(m \neq 1)) = 0$.

Consequently, a stable, nontrivial bcd-parametrization admits feature learning (at least in the last layer activations) if and only if $r = 0$.

Remark 19 (Effective feature learning) *As for perturbations, feature learning in later layers can be caused by weight updates in earlier layers that propagate through the network. One could demand effective feature learning in the l-th layer as $\delta W_t^l x_t^{l-1} = \Theta(1)$ and it would occur if and only if $\min(b_{L+1}, c_{L+1}, d + d_{L+1}) + c_l - \mathbb{I}(l \neq 1) = 0$.*

As for nontriviality, perturbation nontriviality in the output is attained if the constraints for $\tilde{\delta} W_t^{L+1} \tilde{x}_t^L$ or $W_t^l \tilde{\delta} x_t^L$ are exactly satisfied. If perturbations vanish in all layers, SAM's training dynamics collapse to SGD dynamics with scale.

Theorem 20 (Perturbation nontriviality characterization) *Let $l \in [L]$. A stable bcd-parametrization is perturbation nontrivial with respect to the l-th layer if and only if*

$$\tilde{r}_l := \min(b_{L+1}, c_{L+1}) + d + \min_{m=1}^{l}(d_m - \mathbb{I}(m \neq 1)) = 0.$$

A stable bcd-parametrization is perturbation nontrivial with respect to the output if and only if $d + d_{L+1} = 1$ *or* $\min(b_{L+1}, c_{L+1}) + \tilde{r} = 1$.

The converse formulation of the perturbation-nontriviality results characterizes the regime of vanishing perturbations.

Corollary 21 (Vanishing perturbation characterization)

For any $l_0 \in [L]$*, the following statements are equivalent:*

(a) A stable bcd-parametrization has vanishing perturbations in layer l_0.
(b) A stable bcd-parametrization has vanishing perturbations in layer l *for all* $1 \leq l \leq l_0$.
(c) $\tilde{r}_{l_0} := \min(b_{L+1}, c_{L+1}) + d + \min_{m=1}^{l_0}(d_m - \mathbb{I}(m \neq 1)) > 0$.

A stable bcd-parametrization has vanishing perturbations with respect to all layers and the output function if and only if $d_{L+1} > 1/2$ *and* $\tilde{r} > \max(0, 1 - b_{L+1})$. *This case reduces to the results in Yang and Hu [52].*

For the class of stable and perturbation non-trivial bcd-parameterizations, SAM learning is both stable and deviates from SGD dynamics. A natural question to ask here is: what should be the ideal SAM behaviour in the infinite-width limit? To address this question, we make the following crucial distinction between non-vanishing and effective perturbations.

Non-vanishing versus effective perturbations. Recall that the weight perturbation ε^l perturbs the l-th layer's activations as

$$x^l + \tilde{\delta}x^l = \phi((W^l + \varepsilon^l)(x^{l-1} + \tilde{\delta}x^{l-1})),$$

where $\tilde{\delta}x^l$ denotes the perturbation of the l-th layer's activations accumulated from the weight perturbations $\{\varepsilon^{l'}\}_{l' \in [l-1]}$ in all previous layers. Therefore, perturbations $\tilde{\delta}x^l$ can stem both from weight perturbations $\varepsilon^{l'}$ in a previous layer $l' < l$ and/or from weight perturbations ε^l from the current layer l. Intuitively, if we perturb a layer, we want this to affect the next layer's activations and thereby have a non-trivial effect on the output function (i.e., non-vanishing and non-exploding with width). Otherwise one can simply set the layer's perturbations to 0 by design and not change the learning algorithm in the infinite-width limit. This motivates the definition of *effective perturbations*. Without an effective perturbation ε^l of the l-th layer, this layer does not inherit SAM's inductive bias towards low spectral norm of the Hessian or enhanced sparsity and does not improve generalization performance. We provide empirical evidence for these claims in Appendix VII.2. Therefore it is crucial to distinguish between *non-vanishing perturbations* $\tilde{\delta}x_t^l = \Omega(1)$ and *effective perturbations* $\varepsilon_t^l(x_t^{l-1} + \tilde{\delta}x_t^{l-1})) = \Theta(1)$, where $x_t^0 + \tilde{\delta}x_t^0 = \xi_t$.

The following theorem formulates the characterizing conditions under which layers are effectively perturbed.

Theorem 22 (Effective perturbation characterization) *For* $l \in [L]$*, a stable bcd-parametrization effectively performs SAM in the* l*-th layer if and only if* $\min(b_{L+1}, c_{L+1}) + d + d_l - \mathbb{I}(l \neq 1) = 0$.

A stable bcd-parametrization effectively performs SAM in the last layer if and only if $d + d_{L+1} = 1$.

The above understanding of all update and perturbation scalings allows us to extract the most important consequences of different choices of perturbation scaling on the learning dynamics.

Theorem 23 (Global Perturbation Scaling) *Given any stable bcd-parametrization* $\{b_l\}_{l\in[L+1]} \cup \{c_l\}_{l\in[L+1]} \cup \{d_l\}_{l\in[L+1]} \cup \{d\}$. *The parametrization performs updates in the original gradient direction if and only if* $d_l = C$ *for all* $l \in [L+1]$ *for some* $C \in \mathbb{R}$. *In this case, the parametrization has vanishing perturbations in all hidden layers* $l \in [L]$, *and the last layer* $l = L+1$ *is effectively perturbed if and only if* $d = 1/2$. *If* $b_{L+1} > 1/2$ *(as in* μ*P), the gradient norm is dominated by the last layer and simplifies to,*

$$\|v_t\| = \Theta(n^{1/2-C}), \qquad \|v_t\| - L'(f_t(\xi_t), y_t)\|x_t^L\| = o(n^{1/2-C}).$$

One might suspect that it is desirable to let all layers contribute non-vanishingly to the gradient norm in the denominator of (LP). The following proposition shows that this should be avoided with our definition of *bcd*-parameterizations. Of course, if we add even more hyperparameters by decoupling numerator and denominator scalings, we can set all contributions to $\Theta(1)$, which is what we do in Appendix V.8.

Proposition 24 (Balancing gradient norm contributions) *Given any stable bcd-parametrization* $\{b_l\}_{l\in[L+1]} \cup \{c_l\}_{l\in[L+1]} \cup \{d_l\}_{l\in[L+1]} \cup \{d\}$. *If all layers contribute to the gradient norm non-vanishingly in the limit, i.e.* $\|v_t^l\| = \Theta(\|v_t\|)$ *for all* $l \in [L+1], t \in \mathbb{N}_0$, *then the parametrization has vanishing perturbations in all hidden layers* $l \in [L]$. *Such a parametrization effectively performs SAM in the last layer* $l = L+1$ *if and only if* $d = 1/2$.

We argue that for optimal SAM behaviour in the infinite-width limit, every layer of a network must be effectively perturbed. We postulate that just as learning rate transfers across widths under μP due to non-trivial feature evolution with width in all layers, perturbation radius may be transferable across widths if the weights in all layers have non-trivial perturbations with width. Here, we show that there exists a unique stable choice of layerwise perturbation scalings under which a stable *bcd*-parameterization effectively perturbs every single layer. We term this parameterization of the layerwise scalings $\{d_l\}_{l\in[L+1]} \cup \{d\}$ the Maximal Perturbation Parameterization (MPP).

Theorem 25 (Perturbation Scaling Choice for Effective Perturbations) *Given any stable bcd-parametrization* $\{b_l\}_{l\in[L+1]} \cup \{c_l\}_{l\in[L+1]} \cup \{d_l\}_{l\in[L+1]} \cup \{d\}$. *If* $b_{L+1} < 1$, *then there does not exist a stable choice of* $\{d_l\}_{l\in[L+1]} \cup \{d\}$ *that achieves effective perturbations before the last layer. If* $b_{L+1} \geq 1$, *then up to the equivalence* $d_l' = d_l + C$, $C \in \mathbb{R}$, $\forall l \in [L+1]$, *the unique stable choice* $\{d_l\}_{l\in[L+1]} \cup \{d\}$ *with effective perturbations in all layers* $l \in [L+1]$ *is given by*

$$d = -1/2, \qquad d_l = \begin{cases} 1/2 - \min(b_{L+1}, c_{L+1}) & l = 1, \\ 3/2 - \min(b_{L+1}, c_{L+1}) & l \in [2, L], \\ 3/2 & l = L+1. \end{cases} \tag{III.4}$$

Maximal Update and Perturbation Parameterization μP^2. Table III.2 summarizes the consequences of Theorem 25.Theorem 25 provides us with the unique choice of layerwise perturbation

	Definition	SP	SP (stable)	NTP (stable)	μP
b_l	$\mathcal{N}(0, n^{-2b_l})$	$\begin{cases} 0 & l = 1, \\ 1/2 & l \geq 2. \end{cases}$	$\begin{cases} 0 & l = 1, \\ 1/2 & l \geq 2. \end{cases}$	$\begin{cases} 0 & l = 1, \\ 1/2 & l \geq 2. \end{cases}$	$\begin{cases} 0 & l = 1, \\ 1/2 & l \in [2, L], \\ 1, & l = L + 1. \end{cases}$
c_l	LR ηn^{-c_l}	0	1	$\begin{cases} 0 & l = 1, \\ 1 & l \geq 2. \end{cases}$	$\begin{cases} -1 & l = 1, \\ 0 & l \in [2, L], \\ 1 & l = L + 1. \end{cases}$
r	Equation (III.2)	-1	1/2	1/2	0
Stable?			✓	✓	✓
Nontrivial?			✓	✓	✓
Feature learning?					✓

Table III.1: **(bc-parametrizations)** Overview over common implicitly used bc-parametrizations for training MLPs without biases in standard parametrization (SP), standard parametrization with maximal stable nonadaptive LR $c = 1$ (SP (stable)), neural tangent parametrization (NTP) and maximal update parametrization (μP).

	Definition	Naive	Global (stable)	Effective
d	ρn^{-d}	0	$1/2$	$-1/2$
d_l	$n^{-d_l} \nabla_{W^l} L_t$	$1/2$	$1/2$	$\begin{cases} 1/2 - c_\nabla & l = 1, \\ 3/2 - c_\nabla & l \in [2, L], \\ 3/2 & l = L + 1. \end{cases}$
$\tilde{r}$	Equation (III.3)	$c_\nabla - 1/2$	c_∇	0
Stable?		✗	✓	✓
Last layer effectively perturbed?		✗	✓	✓
All layers effectively perturbed?		✗	✗	✓

Table III.2: **(Perturbation scalings)** Overview over important choices of the global perturbation scaling ρn^{-d} and the layerwise perturbation scalings n^{-d_l} for training MLPs without biases with SAM: Naive scaling without width dependence (Naive), maximal stable global scaling along the original gradient direction (Global) and the unique scaling that achieves effective perturbations in all layers (Effective). An extensive overview that characterizes all possible choices of perturbation scaling is provided in Appendix V.3. Recall the gradient scaling $c_\nabla := \min(b_{L+1}, c_{L+1})$.

scaling $\{d_l\}_{l\in[L+1]} \cup \{d\}$ for effective perturbations in all layers. To achieve feature learning in every layer and hyperparameter transfer in the learning rate, μP is the unique[2] choice of layerwise initialization variance and learning rate scalings $\{b_l, c_l\}_{l\in[L+1]}$. Hence, there exists a unique[2] bcd-parameterization that achieves both feature learning and effective perturbations in all layers, we call *maximal update and perturbation parametrization*, μP^2 for short. Note that in μP^2 only the first layer dominates the gradient norm $\|v_t\|$. This shows that a balancing of all layerwise gradient norms should be avoided in the SAM perturbation rule (LP). One could also decouple the numerator and denominator scalings in (LP) and scale the gradient norm contributions of all layers to $\Theta(1)$. The ablations in Appendix VII.4 suggest that this has a negligible effect on the optimal generalization performance, but can be more stable given slightly suboptimal hyperparameters. Now that we have found a parameterization that achieves width-independent scaling of both activation updates and activation perturbations, μP^2 fulfills essential necessary conditions for hyperparameter transfer to occur in both η and ρ.

General perturbation scaling condition. For a generalization to other perturbation rules, we are interested in an intuitive and practical condition of how to scale the perturbations without writing out the NE⊗OR⊤ program. For this purpose we leverage the fact that perturbations $\varepsilon^l \tilde{x}^{l-1}$ and updates $\delta W^l x^{l-1}$ scale very similarly, as long as weight updates δW^l and perturbations ε^l are both gradient-based and correlated with the incoming activations x^{l-1}. Because we set the gradient normalization to $\|v_t\| = \Theta(1)$, we can think of perturbation scalings $n^{-(d+d_l)}$ like we think about learning rate scalings n^{-c_l}. Accordingly, we arrive at the simple scaling condition: weight perturbations of every layer should scale like their updates.

Rule for maximal stable perturbations in μP in the l-th layer:	$\varepsilon_t^l = \Theta(\delta W_t^l)$

Generalizations to other architectures and ASAM variants. This rule can also be applied to other variants of SAM. Table V.2 summarizes its application to two ASAM variants that perform well empirically but cannot be written as a NE⊗OR⊤program. Additional details are provided in Appendix V.5. We demonstrate that these scalings perform well and transfer hyperparameters in Appendix VII. In Appendix V.6, we discuss other architectural components of ResNets and ViTs. According to the perturbation scaling condition (Table V.2), only SAM-ON and elementwise ASAM effectively perturb input-like layers under global scaling in μP. From a scaling perspective, normalization layers behave like input layers and therefore standard applications of SAM or layerwise ASAM do not effectively perturb them. Müller et al. [35, Section 5.3] make the same empirical observations in SP. Irrespective of the SAM variant, our scaling rules even allow to balance perturbations of different layer types and therefore provide precise understanding and control which layers should be perturbed across model scales.

Together with Theorem 25, the following proposition suggests that $b_{L+1} = 1$ is a good choice. However $b_{L+1} > 1$ can also induce effective perturbations, as long as d and d_{L+1} are chosen correctly.

Proposition 26 (Effects of last-layer initialization b_{L+1} on all perturbations) *If a stable bcd-parametrization with $\min(b_{L+1}, c_{L+1}) \leq 1$ is perturbation nontrivial with respect to any hidden layer $l \in [L]$, it is also perturbation nontrivial with respect to the output.*

2. Strictly speaking, unique up to smaller last-layer initialization $b_{L+1} \geq 1$.

Lastly, the following proposition shows that effective perturbations from the first layer propagate through the entire network.

Proposition 27 (Perturbations propagate through the forward pass) *All stable bcd-parametrizations with $d_1 = -\min(b_{L+1}, c_{L+1}) - d$ effectively perturb the first layer and are perturbation nontrivial in all layers.*

Remark 28 (Efficiency gains) *The above results may be used for efficiency gains. Given any stable bcd-parametrization, we can compute the maximal layer l_0 such that $\tilde{r}_{l_0} > 0$, and in wide networks do not have to compute SAM perturbations before layer $l_0 + 1$; as soon as $b_{L+1} > 1/2$ (as for μP), the gradient norm for the SAM update rule is approximately given by $\|\nabla L_t\| \approx L'(f_t(\xi_t), y_t)\|x_t^L\|$, which can directly be computed without an additional backward pass. The practical recommendation from our experiments however is to either use μP^2 or to completely abstain from perturbations.*

Remark 29 (SAM without gradient normalization) *For the SAM update rule without gradient normalization simply set $d = 0$ and remove the gradient norm constraints (III.1) to arrive at the adapted NE⊗OR⊤ program and bcd-constraints. Note that standard parametrization gets even more unstable without dividing by $\|\nabla L\| = \Theta(n^{1/2})$, now requiring $d_{L+1} \geq 1$ for stability. Similar to the previous results, this shows that unawareness of bcd-parametrizations requires strongly scaling down ρ for stability, while vasting computation on vanishing perturbations before the last layer. More details can be found in Appendix V.4.*

IV. Proof of main results

In this section we derive the NE⊗OR⊤ program that corresponds to training a MLP without biases with SAM. For simplicity and clarity of the proof, we prove the one-dimensional case $d_{in} = 1$, $d_{out} = 1$, but an extension to arbitrary but fixed d_{in}, d_{out} is straightforward. Recall Assumption 15 that allows us to apply the Tensor Program Master Theorem and explicitly state the infinite-width limit of training MLPs with SAM in Appendix IV.2.

IV.1. Tensor program formulation

IV.1.1. TENSOR PROGRAM INITIALIZATION

We initialize the matrices $W_0^2, \dots, W_0^L$ as $(W_0^l)_{\alpha\beta} \sim \mathcal{N}(0, 1/n)$, which absorbs $b_l = 1/2$.

We initialize the input layer matrix $W_0^1 \in \mathbb{R}^{n\times 1}$ and normalized output layer matrix $\hat{W}_0^{L+1} = W_0^{L+1} n^{b_{L+1}} \in \mathbb{R}^{1\times n}$ as $(W_0^1)_\alpha, (\hat{W}_0^{L+1})_\alpha \sim \mathcal{N}(0, 1)$, as initial vectors should have a distribution that is $\Theta(1)$.

In the NE⊗OR⊤formulation, we write all quantities as $\theta_z z$, where θ_z denotes their scaling n^C for some $C \in \mathbb{R}$ and z therefore has a $\Theta(1)$ distribution. The stability, nontriviality and feature learning conditions then stem from requiring either $\theta_z \to 0$ or $\theta_z = 1$ depending on z and its desired scale.

IV.1.2. FIRST FORWARD PASS

We denote a definition of a Tensor Program (TP) or NE⊗OR⊤computation as $:=$. Compared to MLPs trained with SGD nothing changes in the first forward pass,

$$h_0^1(\xi) := W_0^1 \xi \quad \text{(NL)}, \quad x_0^l := \phi(h_0^l) \quad \text{(NL)}, \quad h_0^{l+1} := W_0^{l+1} x_0^l. \quad \text{(MatMul)}$$

In the case of MuP, $f_0(\xi) = W_0^{L+1} x_0^L(\xi) \to 0$ defines a scalar in the TP.

Observe the scalings $x_0^1 = \Theta(h_0^1) = \Theta(n^{-b_1})$, $x_0^l = \Theta(h_0^l) = \Theta(n^{1/2-b_l})$ for $l \in [2, L]$ due to CLT, independence at initialization and $x_0^l = \Theta(h_0^l) = \Theta(1)$ by stability. Hence stability at initialization inductively requires $b_1 = 0$, $b_l = 1/2$ for $l \in [2, L]$ and $b_{L+1} \geq 1/2$.

IV.1.3. FIRST BACKWARD PASS

The chain rule of the derivative remains the same, we just evaluate on different weights compared to standard SGD. We denote the adversarially perturbed weights by $\tilde{W}_t^l$ and the normalized perturbations by $\tilde{\delta} W_t^l$. Before computing the updates we have to compute a full backward pass to determine these perturbed weights for each layer, and then compute a forward pass with these perturbed weights to compute the perturbed preactivations $\tilde{h}_t^l$ that we will need for computing the SAM update. Therefore the NE⊗OR⊤ program for SAM maintains a perturbed copy of all preactivations, activations, last-layer weights and logits just for computing the updates of the actual parameters.

Under MuP, the loss derivative with respect to the function remains $\chi_0 := L'(f_0(\xi_0), y_0) \to \mathring{\chi}_0 := L'(0, y_0)$. For the weight perturbation, we need to perform a SGD backward pass,

$$dx_0^L := \hat{W}_0^{L+1}, \quad dh_0^l := dx_0^l \odot \phi'(h_0^l), \quad dx_0^{l-1} := (W_0^l)^T dh_0^l,$$

where $dz := \theta_\nabla^{-1} \nabla_z f$. For SGD (and for SAM, as we will see later) all gradients have scaling $\theta_\nabla := n^{-b_{L+1}}$ in the first step, whereas we overload the notation $\theta_\nabla := n^{-\min(b_{L+1}, c_{L+1})}$ for all later steps. For clarity of presentation assume $b_{L+1} \geq c_{L+1}$ here, the other case follows analogously. For the first step this can be understood from

$$\nabla_{x^L} f_0 = W_0^{L+1} = \Theta(n^{-b_{L+1}}), \qquad \nabla_{h^L} f_0 = \nabla_{x^L} f_0 \odot \phi'(h_0^L) = \Theta(n^{-b_{L+1}}),$$

since $h_0^L = \Theta(1)$ by the stability assumption, and this scale $\Theta(n^{-b_{L+1}})$ propagates through all layers via the chain rule and remains stable in later backward passes. For hidden layer gradients, observe that

$$\begin{aligned}
\nabla_{x^{L-1}} f_t = & \quad (W_t^L)^T \nabla_{h^L} f_t = (W_0^L + \Delta W_t^L)^T \nabla_{h^L} f_t \\
= & \quad \Theta\left((W_0^L)^T \nabla_{h^L} f_t - n^{-c_L} \sum_{s=0}^{t-1} ((\nabla_{h^L} f_s)^T \nabla_{h^L} f_t) x_s^{L-1} \right) \\
= & \quad \Theta(n^{1-2b_L} \theta_\nabla - n^{-c_L} \theta_\nabla^2 n) = \Theta(\theta_\nabla),
\end{aligned}$$

where first term's scale stems from the products $(W_0^L)^T W_0^L v = \Theta(n^{1-2b_L} v)$ due to Yang [51], $b_L = 1/2$ for stability at initialization and $b_{L+1} + c_L \geq 1$ for update stability during training ($r \geq 0$). If we allowed the second term to strictly dominate, the gradient scale would explode iteratively in the backward pass.

The gradient norm. Before computing the weight perturbations, we need to compute the gradient norm for the SAM update. The gradient norm at time t in each layer $l \in [2, L]$ is given by the scalar,

$$\theta_\nabla^{-2} \left\| \frac{\partial L_t}{\partial W^l} \right\|^2 = \sum_{i,j=1}^{n} \left(\chi_t (dh_t^l)_i (x_t^{l-1})_j \right)^2 = \chi_t^2 \|dh_t^l (x_t^{l-1})^T\|_F^2 = \chi_t^2 \big((dh_t^l)^T dh_t^l \big) \big((x_t^{l-1})^T x_t^{l-1} \big),$$

where $\chi_t = L'(f_t(\xi_t), y_t)$ and we used $\partial h^l / \partial W_{ij}^l = (x_j^{l-1} \delta_{ik})_{k=1,\dots,n}$.

Hence the gradient norm of all weights jointly is given by the unnormalized scalar

$$\|\nabla_w L_t\|^2 = \chi_t^2 \left(n\theta_\nabla^2 \frac{(dh_t^1)^T dh_t^1}{n} (\xi_t^T \xi_t) + \sum_{l=2}^{L} n^2 \theta_\nabla^2 \frac{(dh_t^l)^T dh_t^l}{n} \frac{(x_t^{l-1})^T x_t^{l-1}}{n} + n \frac{(x_t^L)^T x_t^L}{n} \right), \tag{IV.1}$$

with scaling $\theta_{\|\nabla\|}^2 = \Theta(n^2\theta_\nabla^2 + n) = \Theta(n)$, because stability at initialization requires $b_{L+1} \geq 1/2$ so that $n^2\theta_\nabla^2 \leq n$. Note that the first layer contributes vanishingly to the gradient norm, the hidden layer gradients only if $b_{L+1} = 1/2$ (equivalently $f_0 = \Theta(1)$) and the last-layer activations always in dominating order. So in μP, in the limit, $\|\nabla_w L_t\| = L'(f_t(\xi_t), y_t)\|x_t^L\|$. This means that the unscaled gradient always aligns with the last-layer activation. For learning in μP, this dominance is corrected by the layerwise learning rates.

The squared norm of the rescaled gradient is given by

$$\begin{aligned} \|v_t\|^2 = \chi_t^2 \Bigg(& n\theta_\nabla^2 n^{-2d_1} \frac{(dh_t^1)^T dh_t^1}{n} (\xi_t^T \xi_t) \\ & + \sum_{l=2}^{L} n^2 \theta_\nabla^2 n^{-2d_l} \frac{(dh_t^l)^T dh_t^l}{n} \frac{(x_t^{l-1})^T x_t^{l-1}}{n} + n n^{-2d_{L+1}} \frac{(x_t^L)^T x_t^L}{n} \Bigg), \end{aligned} \tag{IV.2}$$

with scaling $\theta_v^2 = \Theta(n^{1-2d_1}\theta_\nabla^2 + \sum_{l=2}^{L} n^{2-2d_l}\theta_\nabla^2 + n^{1-2d_{L+1}})$. For simplicity, set $\theta_v = 1$. This raises the constraints $n^{1-2d_1}\theta_\nabla^2 \leq 1$, $n^{2-2d_l}\theta_\nabla^2 \leq 1$ for $l \in [2, L]$ and $n^{1-2d_{L+1}} \leq 1$, which can be rewritten as

$$d_1 \geq 1/2 - \min(b_{L+1}, c_{L+1}), \quad d_l \geq 1 - \min(b_{L+1}, c_{L+1}) \text{ for } l \in [2, L], \quad d_{L+1} \geq 1/2,$$

where at least one equality is demanded to hold in order to attain $\theta_v = 1$. If one of the equalities holds, the respective layer contributes to the norm non-vanishingly in the limit.

Thus, applying the square root and dividing by $\theta_v = 1$ the square root of (IV.2) defines a normalized TP scalar.

Perturbations. Stability implies that also the perturbed (pre-)activations and output function remain $\Theta(1)$ and $O(1)$ respectively. Otherwise a SAM training step would induce blowup in the updates. We call this weaker property of just the perturbations *perturbation stability*.

Definition 30 (Perturbation stability) *We call a bcd-parametrization perturbation stable if and only if $\tilde{h}_t^l, \tilde{x}_t^l = \Theta(1)$ for all $l \in [L]$ and $t \in \mathbb{N}$ and $\tilde{\delta} f_t = O(1)$ for all $t \in \mathbb{N}$.*

Mathematically we get the normalized weight perturbations for $l \in \{2, \dots, L\}$,

$$\tilde{\delta}W_0^{L+1} := \frac{\rho\ \chi_0\ x_0^L}{\|v_0\|}, \quad \tilde{\delta}W_0^l = \frac{\rho\ \chi_0\ dh_0^l\ (x_0^{l-1})^T}{\|v_0\|}, \quad \tilde{\delta}W_0^1 = \frac{\rho\ \chi_0\ dh_0^1\ \xi_0^T}{\|v_0\|},$$

which scale as $\tilde{\theta}_{L+1} := \tilde{\theta}_{W^{L+1}} := n^{-(d+d_{L+1})}$, $\Theta(n^{(d+d_l)-b_{L+1}})$ and $\Theta(n^{-(d+d_1)-b_{L+1}})$ respectively. But the NE$\otimes$OR$\top$ program computation rules do not allow to compute matrices $\tilde{\delta}W_0^l, l \in [L]$, therefore we use the weight updates to directly compute the preactivation and activation changes analogous to the t-th forward pass. For all $t \geq 0$, we write

$$\tilde{h}_t^l = h_t^l + \tilde{\theta}_l \tilde{\delta} h_t^l, \qquad \tilde{x}_t^l = x_t^l + \tilde{\theta}_l \tilde{\delta} x_t^l,$$

with the perturbations for $l \in [2, L]$,

$$\begin{aligned}
\tilde{\delta}h_0^1(\xi) :=& \quad + \frac{\rho\chi_0(\xi_0^T\xi)dh_0^1}{\|v_0\|}, \\
\tilde{\delta}x_t^l :=& \quad \tilde{\theta}_l^{-1}(\phi(h_t^l + \tilde{\theta}_l\tilde{\delta}h_t^l) - \phi(h_t^l)), \\
\tilde{\theta}_l\tilde{\delta}h_0^l :=& \quad \tilde{\theta}_{l-1}W_0^l\tilde{\delta}x_0^{l-1} + (\tilde{W}_0^l - W_0^l)\tilde{x}_0^{l-1} \\
=& \quad \tilde{\theta}_{l-1}W_0^l\tilde{\delta}x_0^{l-1} + \rho\tilde{\theta}_{W^l}\frac{\chi_0}{\|v_0\|}\frac{(x_0^{l-1})^T\tilde{x}_0^{l-1}}{n}dh_0^l,
\end{aligned}$$

which defines a NonLin operation with the vectors $W_0^l\tilde{\delta}x_0^{l-1}$ and dh_0^l and everything else treated as scalars, and with first backward pass scalings $\tilde{\theta}_{W^1} := n^{-(d+d_1)}\theta_\nabla$, $\tilde{\theta}_{W^l} := n^{1-(d+d_l)}\theta_\nabla$ and $\tilde{\theta}_l := \max(\tilde{\theta}_{l-1}, \tilde{\theta}_{W^l}) = \max_{m=1}^l \tilde{\theta}_{W^m}$, where we used that $\tilde{x}_0^{l-1} = \Theta(1)$ due to perturbation stability. Note that these scalings may implicitly increase when $t > 0$ since $\theta_\nabla = n^{-b_{L+1}}$ gets replaced by $\theta_\nabla = n^{-\min(b_{L+1}, c_{L+1})}$.

The activation perturbations can then simply be defined via the NonLin operation,

$$\tilde{\delta}x_0^l := \tilde{\theta}_l^{-1}(\phi(h_0^l + \tilde{\theta}_l\tilde{\delta}h_0^l) - \phi(h_0^l)),$$

with the same scaling as $\tilde{\delta}h_0^l$.

The perturbation of the scalar output function can simply be defined via the NonLin operation,

$$\tilde{\delta}f_0 := \tilde{W}_0^{L+1}\tilde{x}_0^L - W_0^{L+1}x_0^L = \tilde{\theta}'_{L+1}\frac{\tilde{\delta}W_0^{L+1}\tilde{x}_0^L}{n} + \tilde{\theta}'_{L\nabla}\frac{\hat{W}_0^{L+1}\tilde{\delta}x_0^L}{n},$$

with $\tilde{\theta}'_{L+1} := n\tilde{\theta}_{W^{L+1}}$ and $\tilde{\theta}'_{L\nabla} := n\theta_\nabla\tilde{\theta}_L$.

SAM Update. Finally, we can compute the SAM updates as follows. In the case $\min(b_{L+1}, c_{L+1}) \leq d + d_{L+1}$ the weight perturbation scale is dominated by the weight scale, so that

$$dx_{SAM,0}^L := \hat{W}_0^{L+1} + \tilde{\theta}_{(L+1)/\nabla}\ \tilde{\delta}W_0^{L+1},$$

with $\tilde{\theta}_{(L+1)/\nabla} := \tilde{\theta}_{L+1}/\theta_\nabla \leq 1$, whereas if $\min(b_{L+1}, c_{L+1}) > d + d_{L+1}$ we write

$$dx_{SAM,0}^L := \tilde{\theta}_{\nabla/(L+1)}\hat{W}_0^{L+1} + \tilde{\delta}W_0^{L+1},$$

with $\theta_{\nabla/(L+1)} := \theta_\nabla / \tilde{\theta}_{L+1} \leq 1$. In any case, the scaling of $dx^L_{SAM,0}$ and all other SAM gradients is $\theta_{SAM} := \max(\theta_\nabla, n^{-(d+d_{L+1})}) = n^{-\min(b_{L+1}, c_{L+1}, d+d_{L+1})}$. The other SAM gradients are given by

$$\begin{aligned} dh^l_{SAM,0} &:= dx^l_{SAM,0} \odot \phi'(\tilde{h}^l_0) \\ dx^{l-1}_{SAM,0} &:= (\tilde{W}^l_0)^T dh^l_{SAM,0} = (W^l_0 + \tilde{\theta}_{W^l} \tilde{\delta} W^l_0)^T dh^l_{SAM,0} \\ &= (W^l_0)^T dh^l_{SAM,0} + \rho \theta_{SAM} \tilde{\theta}_{W^l} \frac{\chi_0}{\|v_0\|} \frac{(dh^l_0)^T dh^l_{SAM,0}}{n} x^{l-1}_0. \end{aligned}$$

where the last line define a NonLin operation in the vectors $(W^l_0)^T dh^l_{SAM,t}$ and x^{l-1}_0 and everything else treated as scalars. Consequently, $\nabla_{h^l_0} f|_{\tilde{W}_0}$ is of the same scale as $\nabla_{x^l_0} f|_{\tilde{W}_0}$ and $\nabla_{x^{l-1}_0} f|_{\tilde{W}_0}$ is of the scale $\max(\theta_{SAM}, \tilde{\theta}_{W^l} \theta_{SAM}) = \theta_{SAM}$ since $\tilde{\theta}_{W^l} \leq 1$ is required for perturbation stability.

Note that for SAM's weight updates the loss derivative is also evaluated on the perturbed weights,

$$\tilde{\chi}_0 := L'(\tilde{W}^{L+1}_0 \tilde{x}^L_0, y_0).$$

Constraints on the output function. Assuming $\tilde{x}^L_0 = \Theta(1)$ (perturbation stability), we get $\tilde{\chi}_0 = O(1)$ if and only if $\tilde{\delta} W^{L+1}_0 = O(n^{-1})$ if and only if $d + d_{L+1} \geq 1$.

We have $\tilde{\chi}_0 = \Theta(1)$ if and only if $\tilde{f}_0 = \tilde{W}^{L+1}_0 \tilde{x}^L_0 = \Theta(1)$. This can either be caused by changes in the last-layer weights, by non-vanishing initial function $W^{L+1}_0 x^L_0$ (if and only if $b_{L+1} = 1/2$) or by $W^{L+1}_0 \tilde{\delta} x^L_0 = \Theta(1)$, which holds if and only if $b_{L+1} + \tilde{r}_L = 1$ (analogously, $W^{L+1}_0 \tilde{\delta} x^L_0 = O(1)$ if and only if $b_{L+1} + \tilde{r}_L \geq 1$). The first case requires $\tilde{\delta} W^{L+1}_0 = \Theta(n^{-1})$, since $\tilde{\delta} W^{L+1}_0$ and $\tilde{x}^L_0$ are highly correlated. $\tilde{\delta} W^{L+1}_0 = \Theta(n^{-1})$ is fulfilled if and only if $d + d_{L+1} = 1$ (the analogue to $c_{L+1} \geq 1$ for stability and $c_{L+1} = 1$ for nontriviality).

Hence perturbation stability of the output function holds only if $d + d_{L+1} \geq 1$ and $b_{L+1} + \tilde{r}_L \geq 1$. Then, perturbation nontriviality holds if and only if $d + d_{L+1} = 1$ or $b_{L+1} + \tilde{r}_L = 1$.

In the t-th backward pass, $b_{L+1} + \tilde{r}_L \geq 1$ will be replaced by the slightly stronger constraint $b_{L+1} + \tilde{r} \geq 1$.

IV.1.4. t-TH FORWARD PASS

Formally, we sum the updates in each step,

$$\hat{W}^{L+1}_t := \hat{W}^{L+1}_0 + \theta_{L+1/\nabla}(\delta W^{L+1}_1 + \cdots + \delta W^{L+1}_t),$$

where $\delta W^{L+1}_{t+1} := -\eta \ \tilde{\chi}_t \ (\tilde{x}^L_t)^T$ denotes the normalized change in the weights W^{L+1} (as a row vector) of scaling $\theta_{L+1} = \theta_{W^{L+1}} = n^{-c_{L+1}}$ under perturbation stability and nontriviality so that $\hat{W}^{L+1}_t$ scales as $\theta_\nabla = n^{-\min(b_{L+1}, c_{L+1})}$. δW^{L+1}_{t+1} should not be confused with $\tilde{\delta} W^{L+1}_{t+1}$ which denotes the perturbation of the weights at time $t+1$. For every nontrivial stable parametrization we have $\tilde{\chi}_t = \Theta(1)$ and $\tilde{x}^L_t = \Theta(1)$ which requires $\tilde{\theta}_L \leq 1$. In the case $c_{L+1} < b_{L+1}$, we write $\hat{W}^{L+1}_t := n^{-b_{L+1}+c_{L+1}} \hat{W}^{L+1}_0 + (\delta W^{L+1}_1 + \cdots + \delta W^{L+1}_t)$ with the same scaling $\theta_\nabla = n^{-\min(b_{L+1}, c_{L+1})}$.

For preactivations and activations we also sum the changes from each step,

$$h^l_t := h^l_0 + \theta_l(\delta h^l_1 + \cdots + \delta h^l_t), \qquad x^l_t := x^l_0 + \theta_l(\delta x^l_1 + \cdots + \delta x^l_t).$$

Using the fact that

$$W_t^1 - W_{t-1}^1 = -\eta\tilde{\chi}_{t-1}\theta_{W^1} dh_{SAM,t-1}^1 \xi_{t-1}^T,$$

yields the normalized preactivation updates

$$\delta h_t^1(\xi) := -\eta\tilde{\chi}_{t-1} dh_{SAM,t-1}^1 \xi_{t-1}^T \xi \quad \text{(NL)},$$

with scaling $\theta_1 = \theta_{W^1} = n^{-c_1}\theta_{SAM} = n^{-c_1 - \min(b_{L+1}, c_{L+1}, d+d_{L+1})}$ as for SGD under perturbation stability and nontriviality where $\tilde{\chi}_{t-1} = \Theta(1)$.

For $l \in [2, L]$, it holds that

$$W_t^l - W_{t-1}^l = -\eta\tilde{\chi}_{t-1}\theta_{W^l}\frac{1}{n} dh_{SAM,t-1}^l (\tilde{x}_{t-1}^{l-1})^T,$$

with the right scaling $\theta_{W^l} = n^{1-c_l - \min(b_{L+1}, c_{L+1}, d+d_{L+1})}$ as for SGD under perturbation stability $\tilde{x}_{t-1}^{l-1} = \Theta(1)$, so that we get δh_t^l using a telescope sum,

$$\begin{aligned}
\theta_l \delta h_t^l = \quad & W_t^l x_t^{l-1} - W_{t-1}^l x_{t-1}^{l-1} = W_{t-1}^l (x_t^{l-1} - x_{t-1}^{l-1}) + (W_t^l - W_{t-1}^l) x_t^{l-1} \\
= \quad & \theta_{l-1}\left(W_0^l \delta x_t^{l-1} + \sum_{s=1}^{t-1} (W_s^l - W_{s-1}^l)\delta x_t^{l-1} \right) + (W_t^l - W_{t-1}^l) x_t^{l-1} \\
= \quad & \theta_{l-1}\left(W_0^l \delta x_t^{l-1} - \eta\theta_{W^l} \sum_{s=1}^{t-1} \tilde{\chi}_{s-1} \frac{(\tilde{x}_{s-1}^{l-1})^T \delta x_t^{l-1}}{n} dh_{SAM,s-1}^l \right) \\
& - \eta\theta_{W^l}\tilde{\chi}_{t-1}\frac{(\tilde{x}_{t-1}^{l-1})^T x_t^{l-1}}{n} dh_{SAM,t-1}^l,
\end{aligned}$$

which defines a NonLin operation with the vectors $W_0^l \delta x_t^{l-1}, dh_{SAM,0}^l, dh_{SAM,t-1}^l$ and everything else treated as scalars. The scaling is given by

$$\theta_l = \max(\theta_{l-1}, \theta_{W^l}\theta_{l-1}, \theta_{W^l}) = \max_{m=1}^{l} \theta_{W^m} = n^{-r_l},$$

with

$$r_l := \min(b_{L+1}, c_{L+1}, d + d_{L+1}) + \min_{m=1}^{l}(c_m - \mathbb{I}(m \neq 1)),$$

where $\theta_{W^l} \leq 1$ for all $l \in [L]$ for stability. Note that for $l_1 \leq l_2$, it holds that $\theta_{l_1} \leq \theta_{l_2}$, which explains the sufficiency of $\theta_L = n^{-r_L} = n^{-r}$ for the stability of the activation updates.

Activations with the same scaling θ_l can then simply be defined via the NonLin operation

$$\delta x_t^l := \theta_l^{-1}(\phi(h_{t-1}^l + \theta_l \delta h_t^l) - \phi(h_{t-1}^l)).$$

The updates of the output function are scalars defined as

$$\delta f_t := \theta'_{L+1}\frac{\delta W_t^{L+1} x_t^L}{n} + \theta'_{L\nabla}\frac{\hat{W}_{t-1}^{L+1}\delta x_t^L}{n},$$

where $\theta'_{L+1} = n\theta_{L+1} = n^{1-c_{L+1}}$ and $\theta'_{L\nabla} = n\theta_\nabla\theta_L = n^{1-\min(b_{L+1}, c_{L+1}) - r_L}$, where we will see why $W_{t-1}^{L+1} = \Theta(n^{-\min(b_{L+1}, c_{L+1})})$ in the next paragraph. This leads to the constraints $c_{L+1} \geq 1$ and $b_{L+1} + r \geq 1$ for the stability of the output function, where equality in either constraint leads to nontriviality.

IV.1.5. t-TH BACKWARD PASS

Perturbations. Due to linearity and stability, the last layer remains

$$dx_t^L := \hat{W}_t^{L+1},$$

with scaling $\theta_\nabla = n^{-\min(b_{L+1}, c_{L+1})}$.

As in the first backward pass, we use the weight updates to directly compute the preactivation and activation perturbations similar to the t-th forward pass but performing SGD instead of SAM in the last step. The SGD backward pass for the perturbation is given by

$$\begin{aligned}
dh_t^l :=& \quad dx_t^l \odot \phi'(h_t^l),\\
dx_t^{l-1} :=& \quad (W_t^l)^T dh_t^l\\
=& \quad \left(W_0^l - \eta\theta_{W^l}\sum_{s=1}^{t}\tilde{\chi}_{s-1}\frac{1}{n}dh_{SAM,s-1}^l(\tilde{x}_{s-1}^{l-1})^T\right)^T dh_t^l\\
=& \quad W_0^l dh_t^l - \eta(n^{1-c_l}\theta_{SAM}\theta_\nabla)\sum_{s=1}^{t}\tilde{\chi}_{s-1}\frac{(dh_{SAM,s-1}^l)^T dh_t^l}{n}\tilde{x}_{s-1}^{l-1},
\end{aligned}$$

with scaling $\max(\theta_\nabla, n^{1-c_l}\theta_{SAM}\theta_\nabla) = \theta_\nabla$, since $n^{1-c_l}\theta_{SAM} \leq 1$ is implied by $r \geq 0$ required for the stability of (pre-)activation updates.

We write $\chi_t = L'(f_t(\xi_t), y_t)$ for the derivative of the loss with respect to the unperturbed function (which is $\Theta(1)$ under stability and nontriviality), and get

$$\begin{aligned}
\tilde{\delta}h_t^1(\xi) :=& \quad +\frac{\rho\chi_t(\xi_t^T\xi)dh_t^1}{\|v_t\|},\\
\tilde{\theta}_l\tilde{\delta}h_t^l :=& \quad \tilde{\theta}_{l-1}W_t^l\tilde{\delta}x_t^{l-1} + (\tilde{W}_t^l - W_t^l)\tilde{x}_t^{l-1}\\
=& \quad \tilde{\theta}_{l-1}\left(W_0^l\tilde{\delta}x_t^{l-1} + \sum_{s=1}^{t}(W_s^l - W_{s-1}^l)\tilde{\delta}x_t^{l-1}\right) + (\tilde{W}_t^l - W_t^l)\tilde{x}_t^{l-1}\\
=& \quad \tilde{\theta}_{l-1}\left(W_0^l\tilde{\delta}x_t^{l-1} - \eta(n^{1-c_l}\theta_{SAM})\sum_{s=1}^{t}\tilde{\chi}_{s-1}\frac{(\tilde{x}_{s-1}^{l-1})^T\tilde{\delta}x_t^{l-1}}{n}dh_{SAM,s-1}^l\right)\\
&+\rho\tilde{\theta}_{W^l}\frac{\chi_t}{\|v_t\|}\frac{(x_t^{l-1})^T\tilde{x}_t^{l-1}}{n}dh_t^l,
\end{aligned}$$

which defines a NonLin operation with the vectors $W_0^l\tilde{\delta}x_t^{l-1}, dh_{SAM,0}^l, \ldots, dh_{SAM,t-1}^l, dh_t^l$, and where we can now define the definitive scalings $\tilde{\theta}_1 := \tilde{\theta}_{W^1} := n^{-(d+d_1)}\theta_\nabla = n^{-(\min(b_{L+1},c_{L+1})+d+d_1)}$, $\tilde{\theta}_{W^l} := n^{1-(d+d_l)}\theta_\nabla = n^{-(\min(b_{L+1},c_{L+1})+d+(d_l-1))}$ and $\tilde{\theta}_l = \max(\tilde{\theta}_{l-1}, n^{1-c_l}\theta_{SAM}\tilde{\theta}_{l-1}, \tilde{\theta}_{W^l}) = \max_{m=1}^{l}\tilde{\theta}_{W^m} = n^{-\tilde{r}_l}$ with

$$\tilde{r}_l := \min(b_{L+1}, c_{L+1}) + d + \min_{m=1}^{l}(d_m - \mathbb{I}(m \neq 1)),$$

where we used that $n^{1-c_l}\theta_{SAM} \leq 1$ due to $r \geq 0$ for stability and $\tilde{x}_t^{l-1} = \Theta(1)$ due to perturbation stability. Perturbation stability of the hidden layer (pre-)activations $\tilde{\delta}h^l, \tilde{\delta}x^l = O(1)$ for all $l \in [L]$ holds if and only if $\tilde{r} := \tilde{r}_L \geq 0$ since $\tilde{r}_l \geq \tilde{r}_L$ for all $l \leq L$.

The activation perturbations $\tilde{\delta}x_t^l$ and the perturbation of the output function $\tilde{\delta}f_t$ can be defined exactly as in the first backward pass,

$$\begin{aligned}\tilde{\delta}x_t^l &:= \tilde{\theta}_l^{-1}(\phi(h_t^l + \tilde{\theta}_l\tilde{\delta}h_t^l) - \phi(h_t^l)),\\ \tilde{\delta}f_t &:= \tilde{W}_t^{L+1}\tilde{x}_t^L - W_t^{L+1}x_t^L = \tilde{\theta}'_{L+1}\frac{\tilde{\delta}W_t^{L+1}\tilde{x}_t^L}{n} + \tilde{\theta}'_{L\nabla}\frac{\hat{W}_t^{L+1}\tilde{\delta}x_t^L}{n},\end{aligned}$$

with $\tilde{\delta}W_t^{L+1} := \frac{\rho\,\chi_t\,x_t^L}{\|v_t\|}$ and the same scalings $\tilde{\theta}_l$, $\tilde{\theta}'_{L+1} = n^{1-(d+d_{L+1})}$ and $\tilde{\theta}'_{L\nabla} = n\theta_\nabla\tilde{\theta}_L = n^{1-\min(b_{L+1},c_{L+1})-\tilde{r}}$ since $W_t^{L+1} = W_0^{L+1} + \Delta W_t^{L+1} = \max(n^{-b_{L+1}}, n^{-c_{L+1}})$, which yields the slightly stronger constraint (than in the first backward pass) $\min(b_{L+1}, c_{L+1})+\tilde{r} \geq 1$ for perturbation stability and either $\tilde{\theta}'_{L+1} = 1$ or $\min(b_{L+1}, c_{L+1}) + \tilde{r} = 1$ for perturbation nontriviality.

SAM Update. For each $l \in \{1, \ldots, L\}$, as in the first backward pass, we get

$$dx_{SAM,t}^L := \hat{W}_t^{L+1} + \tilde{\theta}_{(L+1)/\nabla}\,\tilde{\delta}W_t^{L+1},$$

with scaling $\theta_{SAM} = n^{-\min(b_{L+1},c_{L+1},d_{L+1}+1/2)}$ as well as

$$dh_{SAM,t}^l := dx_{SAM,t}^l \odot \phi'(\tilde{h}_t^l).$$

For $dx_{SAM,t}^l$ we again use a telescope sum over the weight changes,

$$\begin{aligned}dx_{SAM,t}^{l-1} &:= (\tilde{W}_t^l)^T dh_{SAM,t}^l = (W_0^l + \theta_{W^l}\sum_{s=1}^t \delta W_s^l + \tilde{\theta}_{W^l}\tilde{\delta}W_t^l)^T dh_{SAM,t}^l\\ &= (W_0^l)^T dh_{SAM,t}^l - \eta(n^{1-c_l}\theta_{SAM})\sum_{s=1}^t \tilde{\chi}_{s-1}\frac{(dh_{SAM,s-1}^l)^T dh_{SAM,t}^l}{n}\tilde{x}_{s-1}^{l-1}\\ &\quad + \rho(n^{1/2-d_l}\theta_\nabla)\frac{\chi_t}{\|v_t\|}\frac{(dh_t^l)^T dh_{SAM,t}^l}{n}x_t^{l-1},\end{aligned}$$

which defines a NonLin operation in the vectors $(W_0^l)^T dh_{SAM,t}^l, \tilde{x}_0^{l-1}, \ldots, \tilde{x}_{t-1}^{l-1}, x_t^{l-1}$ and everything else treated as scalars. Note that the scalings remain θ_{SAM}, since

$$\nabla_{x_t^{l-1}} f|_{\tilde{W}_t} = \Theta(\max(\theta_{SAM}, n^{1-c_l}\theta_{SAM}^2, n^{1/2-d_l}\theta_\nabla\theta_{SAM})) = \Theta(\theta_{SAM})$$

under stability, nontriviality, perturbation stability and perturbation nontriviality.

Finally define the loss derivative on the perturbed output function

$$\tilde{\chi}_t := L'(\tilde{W}_t^{L+1}\tilde{x}_t^L, y_t),$$

and compute the normalized change in W^{L+1},

$$\delta W_{t+1}^{L+1} := -\eta\tilde{\chi}_t\tilde{x}_t^L.$$

IV.2. The infinite-width limit

In this section, we apply the Master Theorem's computation rules to derive the marginal distributions Z corresponding to the vectors of the program constructed above. According to the Master Theorem, each such vector z will have roughly iid coordinates distributed like Z^z in the large n limit.

We assume stability holds, so that $\theta \to \mathring{\theta} \in \{0, 1\}$ for all scalars θ in the program.

For the first forward pass, we have

$$Z^{h_0^1(\xi)} = \xi Z^{W_0^1}, \qquad Z^{x_0^l(\xi)} = \phi(Z^{h_0^l(\xi)}), \qquad Z^{h_0^{l+1}(\xi)} = Z^{W_0^{l+1} x_0^l(\xi)}.$$

If $b_{L+1} > 1/2$ then $\mathring{f}_0 = 0$, otherwise if $b_{L+1} = 1/2$ then $\mathring{f}_0$ converges to a nontrivial Gaussian. For the details we refer to Appendix H.4.1 in Yang and Hu [52], as at initialization their results still hold here.

For the first SGD backward pass, we have

$$Z^{dx_0^L(\xi)} = Z^{\hat{W}_0^{L+1}}, \qquad Z^{dh_0^l(\xi)} = Z^{dx_0^l(\xi)} \phi'(Z^{h_0^l(\xi)}), \qquad Z^{dx_0^{l-1}(\xi)} = Z^{(W_0^l)^T dh_0^l(\xi)},$$

where $\dot{Z}^{dx_0^l(\xi)} = 0$ and $Z^{dx_0^l(\xi)} = \hat{Z}^{dx_0^l(\xi)}$ for all $\xi \in \mathcal{X}$.

For general $t > 0$, we have

$$\begin{aligned}
Z^{dx_t^L(\xi)} &= Z^{\hat{W}_t^{L+1}}, \\
Z^{dh_t^l(\xi)} &= Z^{dx_t^l(\xi)} \phi'(Z^{h_t^l(\xi)}), \\
Z^{dx_t^{l-1}(\xi)} &= Z^{(W_0^l)^T dh_t^l(\xi)} - \eta \mathring{\theta}_{W^l} \sum_{s=1}^{t} \mathring{\tilde{\chi}}_{s-1} \mathbb{E}[Z^{dh_{SAM,s-1}^l} Z^{dh_t^l}] Z^{\tilde{x}_{s-1}^{l-1}},
\end{aligned}$$

where $\mathring{\tilde{\chi}}_s = L'(\mathring{\tilde{f}}_s(\xi_s), y_s)$ for $s < t$, and $Z^{(W_0^l)^T dh_t^l(\xi)}$ is a $\Theta(1)$ random variable distributed as

$$Z^{(W_0^l)^T dh_t^l(\xi)} = \hat{Z}^{(W_0^l)^T dh_t^l(\xi)} + \sum_{v \in \mathcal{V}:\, W_0^l v \in \mathcal{V}} Z^v \, \mathbb{E} \frac{\partial Z^{dh_t^l(\xi)}}{\partial \hat{Z}^{W_0^l v}}.$$

For all $t \geq 0$, the limit of the gradient norm is given by

$$\|\mathring{v}\| = \mathring{\chi}_t \left(\mathring{\theta}^2_{\|v^1\|} \mathbb{E}[Z^{(dh_t^1)^2}](\xi_t^T \xi_t) + \sum_{l=2}^{L} \mathring{\theta}^2_{\|v^l\|} \mathbb{E}[Z^{(dh_t^l)^2}] \mathbb{E}[Z^{(x_t^{l-1})^2}] + \mathring{\theta}^2_{\|v^{L+1}\|} \frac{(x_t^L)^T x_t^L}{n} \right)^{1/2},$$

where $\mathring{\chi}_t = L'(\mathring{f}_t(\xi_t), y_t)$, $\theta^2_{\|v^1\|} := n^{1-2d_1} \theta^2_\nabla$, $\theta^2_{\|v^l\|} := n^{2-2d_l} \theta^2_\nabla$ for $l \in [2, L]$ and $\theta^2_{\|v^{L+1}\|} := n^{1-2d_{L+1}}$, and where $\mathring{\theta}^2_{\|v^{L+1}\|} = 1$ if and only if $d_{L+1} = 1/2$ and $\mathring{\theta}^2_{\|v^{L+1}\|} = 0$ if and only if $d_{L+1} > 1/2$, while $\mathring{\theta}^2_{\|v^l\|} = 1$ if and only if $2d_l = 1 + \mathbb{I}(l > 1) - 2\min(b_{L+1}, c_{L+1})$ and $\mathring{\theta}^2_{\|v^l\|} = 0$ if and only if $2d_l > 1 + \mathbb{I}(l > 1) - 2\min(b_{L+1}, c_{L+1})$.

For the last-layer weight perturbations (for $\theta_\nabla \geq \tilde{\theta}_{L+1}$, else $Z^{\hat{\tilde{W}}_t^{L+1}} = Z^{\tilde{\delta} W_t^{L+1}}$) we have

$$Z^{\hat{\tilde{W}}_t^{L+1}} = Z^{\hat{W}_t^{L+1}} + \mathring{\tilde{\theta}}_{(L+1)/\nabla} Z^{\tilde{\delta} W_t^{L+1}}, \qquad Z^{\tilde{\delta} W_t^{L+1}} = \frac{\rho \, \mathring{\chi}_t}{\|\mathring{v}\|} Z^{x_t^L}.$$

Note that $\mathring{\chi}_t$ cancels itself out and we purely get a perturbation in distribution $Z^{x_t^L}$ scaled to have standard deviation ρ.

For all $t \geq 0$ and $l \in [1, L]$, we have

$$Z^{\tilde{h}_t^l} = Z^{h_t^l} + \mathring{\tilde{\theta}}_l Z^{\tilde{\delta} h_t^l}, \qquad Z^{\tilde{x}_t^l} = Z^{x_t^l} + \mathring{\tilde{\theta}}_l Z^{\tilde{\delta} x_t^l},$$

where for $l = 1$,

$$Z^{\tilde{\delta} h_t^1(\xi)} = + \frac{\rho \mathring{\chi}_t (\xi_t^T \xi)}{\|\mathring{v}\|} Z^{dh_t^1}.$$

If $\mathring{\tilde{\theta}}_l = 0$, then

$$Z^{\tilde{\delta} x_t^l} = \phi'(Z^{h_t^l}) Z^{\tilde{\delta} h_t^l},$$

otherwise $\mathring{\tilde{\theta}}_l = 1$ and

$$Z^{\tilde{\delta} x_t^l} = \phi(Z^{\tilde{h}_t^l}) - \phi(Z^{h_t^l}).$$

For $l \geq 2$, we have

$$\begin{aligned} Z^{\tilde{\delta} h_t^l} = \quad & \mathring{\tilde{\theta}}_{(l-1)/l} \, Z^{W_0^l \tilde{\delta} x_t^{l-1}} - \eta \mathring{\theta}_{W^l(\tilde{l}-1)/\tilde{l}} \sum_{s=1}^{t} \mathring{\tilde{\chi}}_{s-1} \mathbb{E}[Z^{\tilde{x}_{s-1}^{l-1}} Z^{\tilde{\delta} x_t^{l-1}}] Z^{dh_{SAM,s-1}^l} \\ & + \rho \mathring{\tilde{\theta}}_{W^l/l} \frac{\mathring{\chi}_t}{\|\mathring{v}\|} \mathbb{E}[Z^{x_t^{l-1}} Z^{\tilde{x}_t^{l-1}}] Z^{dh_t^l}, \end{aligned}$$

where $\tilde{\theta}_{(l-1)/l} = \frac{\tilde{\theta}_{l-1}}{\tilde{\theta}_l}$, $\theta_{W^l(\tilde{l}-1)/\tilde{l}} = \frac{\theta_{W^l} \tilde{\theta}_{l-1}}{\tilde{\theta}_l}$ and $\tilde{\theta}_{W^l/l} = \frac{\tilde{\theta}_{W^l}}{\tilde{\theta}_l}$, and $Z^{W_0^l \tilde{\delta} x_t^{l-1}}$ has the decomposition

$$Z^{W_0^l \tilde{\delta} x_t^{l-1}} = \hat{Z}^{W_0^l \tilde{\delta} x_t^{l-1}} + \sum_{v \in \mathcal{V}:\ (W_0^l)^T v \in \mathcal{V}} Z^v \, \mathbb{E} \frac{\partial Z^{\tilde{\delta} x_t^{l-1}}}{\partial \hat{Z}^{(W_0^l)^T v}}.$$

The perturbed output function has the limit $\mathring{\tilde{f}}_t := \mathring{f}_t + \mathring{\tilde{\delta}} f_t$ with

$$\mathring{\tilde{\delta}} f_t := \mathring{\tilde{\theta}}'_{L+1} \mathbb{E}[Z^{\tilde{\delta} W_t^{L+1}} Z^{\tilde{x}_t^L}] + \mathring{\tilde{\theta}}'_{L\nabla} \mathbb{E}[Z^{\hat{W}_t^{L+1}} Z^{\tilde{\delta} x_t^L}],$$

so that we can define $\mathring{\tilde{\chi}}_t = L'(\mathring{\tilde{f}}_t(\xi_t), y_t)$ or equivalently $\mathring{\tilde{\chi}}_t = L'(\mathring{\tilde{\theta}}_{L+1} \mathring{\tilde{\theta}}_L \mathbb{E}[Z^{\hat{\tilde{W}}_t^{L+1}} Z^{\tilde{x}_t^L}], y_t)$.

For the SAM gradients we have

$$\begin{aligned} Z^{dx_{SAM,t}^L} = \quad & Z^{\hat{W}_t^{L+1}} + \mathring{\tilde{\theta}}_{(L+1)/\nabla} \, Z^{\tilde{\delta} W_t^{L+1}}, \\ Z^{dh_{SAM,t}^l} = \quad & Z^{dx_{SAM,t}^l} \cdot \phi'(Z^{\tilde{h}_t^l}) \\ Z^{dx_{SAM,t}^{l-1}} = \quad & Z^{(W_0^l)^T dh_{SAM,t}^l} - \eta \mathring{\theta}_{W^l} \sum_{s=1}^{t} \mathring{\tilde{\chi}}_{s-1} \mathbb{E}[Z^{dh_{SAM,s-1}^l} Z^{dh_{SAM,t}^l}] Z^{\tilde{x}_{s-1}^{l-1}} \\ & + \rho \mathring{\tilde{\theta}}_{W^l} \frac{\mathring{\chi}_t}{\|\mathring{v}\|} \mathbb{E}[Z^{dh_t^l} Z^{dh_{SAM,t}^l}] Z^{x_t^{l-1}}, \end{aligned}$$

where $Z^{(W_0^l)^T dh^l_{SAM,t}}$ is given by

$$Z^{(W_0^l)^T dh^l_{SAM,t}} = \hat{Z}^{(W_0^l)^T dh^l_{SAM,t}} + \sum_{v\in\mathcal{V}:\, W_0^l v\in\mathcal{V}} Z^v \, \mathbb{E}\frac{\partial Z^{dh^l_{SAM,t}}}{\partial \hat{Z}^{W_0^l v}}.$$

Now SAM's (pre-)activation updates are given by

$$Z^{h^l_t} = Z^{h^l_0} + \mathring{\theta}_l(Z^{\delta h^l_1} + \cdots + Z^{\delta h^l_t}), \qquad Z^{x^l_t} = Z^{x^l_0} + \mathring{\theta}_l(Z^{\delta x^l_1} + \cdots + Z^{\delta x^l_t}),$$

with, for $l \in [2, L]$,

$$\begin{aligned} Z^{\delta h^1_t(\xi)} = {} & -\eta \mathring{\tilde{\chi}}_{t-1}(\xi^T_{t-1}\xi) Z^{dh^1_{SAM,t-1}}, \\ Z^{\delta h^l_t} = {} & \mathring{\theta}_{(l-1)/l}\left(Z^{W_0^l \delta x^{l-1}_t} - \eta \mathring{\theta}_{W^l} \sum_{s=1}^{t-1} \mathring{\tilde{\chi}}_{s-1} \mathbb{E}[Z^{\tilde{x}^{l-1}_{s-1}} Z^{\delta x^{l-1}_t}] Z^{dh^l_{SAM,s-1}}\right) \\ & - \eta \mathring{\theta}_{W^l/l} \mathring{\tilde{\chi}}_{t-1} \mathbb{E}[Z^{\tilde{x}^{l-1}_{t-1}} Z^{x^{l-1}_t}] Z^{dh^l_{SAM,t-1}}, \end{aligned}$$

where $\theta_{(l-1)/l} := \theta_{l-1}/\theta_l$, $\theta_{W^l/l} := \theta_{W^l}/\theta_l$ and $Z^{W_0^l \delta x^{l-1}_t}$ has the decomposition

$$Z^{W_0^l \delta x^{l-1}_t} = \hat{Z}^{W_0^l \delta x^{l-1}_t} + \sum_{v\in\mathcal{V}:\, (W_0^l)^T v\in\mathcal{V}} Z^v \, \mathbb{E}\frac{\partial Z^{\delta x^{l-1}_t}}{\partial \hat{Z}^{(W_0^l)^T v}}.$$

If $\mathring{\theta}_l = 0$, then

$$Z^{\delta x^l_t} = \phi'(Z^{h^l_{t-1}}) Z^{\delta h^l_t},$$

otherwise $\mathring{\theta}_l = 1$ and

$$Z^{\delta x^l_t} = \phi(Z^{h^l_t}) - \phi(Z^{h^l_{t-1}}).$$

The last-layer SAM weight update is given by

$$Z^{\hat{W}^{L+1}_t} = Z^{\hat{W}^{L+1}_0} + \mathring{\theta}_{L+1/\nabla}(Z^{\delta W^{L+1}_1} + \cdots + Z^{\delta W^{L+1}_t}),$$

with $Z^{\delta W^{L+1}_t} = -\eta \mathring{\tilde{\chi}}_{t-1} Z^{\tilde{x}^L_{t-1}}$.

For $t > 0$, the SAM function update is given by

$$\mathring{f}_t = \mathring{f}_0 + \mathring{\delta} f_1 + \cdots + \mathring{\delta} f_t,$$

with $\mathring{\delta} f_t = \mathring{\theta}'_{L+1} \mathbb{E}[Z^{\delta W^{L+1}_t} Z^{x^L_t}] + \mathring{\theta}'_{L\nabla} \mathbb{E}[Z^{\hat{W}^{L+1}_{t-1}} Z^{\delta x^L_t}]$.

IV.3. Concluding the proof of all main results

After writing out the NE⊗OR⊤program and its limit, as well as tracking all scalings, the main results stated in Appendix III all follow from the Tensor Program Master Theorem and from the characterization results in Yang and Hu [52] in the following way.

Formally Yang and Hu [52] show feature learning for SGD with small enough learning rate $\eta > 0$ by proving $\partial_\eta^2 \mathbb{E}(Z^{x_1^L(\xi_0)})^2 \neq 0$ at $\eta = 0$, and they show that learning does not occur in the kernel regime by showing $\partial_\eta^3 \mathring{f}_1 \neq 0$, hence $\mathring{f}_1 - \mathring{f}_0$ is not linear in η.

Both $\mathbb{E}(Z^{x_1^L(\xi_0)})^2$ and $\mathring{f}_1$ are defined via NE$\otimes$OR$\top$computations and can be written as a composition of additions, multiplications, the expectation operator, applications of ϕ and ϕ', overall applications of infinitely differentiable, pseudo-Lipschitz functions to (Gaussian) random variables, η and ρ. Consequently $\mathbb{E}(Z^{x_1^L(\xi_0)})^2$ and $\mathring{f}_1$ are infinitely often differentiable as a function of both η and ρ, where differentiating the expectation operator is covered in Yang and Hu [52, Lemma H.39]. Since Yang and Hu [52] cover the case $\rho = 0$, their proofs immediately show the correctness of the derived scalings for SAM as long as $\eta > 0$ and $\rho > 0$ are chosen small enough. Both the gradient evaluation for the perturbation as well as the gradient evaluation for the updates stay arbitrarily close to those of SGD if $\rho > 0$ is chosen small enough. The conditions for stability, nontriviality, feature learning, perturbation nontriviality and effective perturbations now follow from considering the respective scaling.

IV.3.1. PROOF OF THEOREM 16

A *bcd*-parameterization is stable if and only if all scalings in the Tensor Program have the limit $\mathring{\theta} \in \{0, 1\}$, where $\mathring{\theta} = 1$ is required for activations at initialization (for which nothing changes compared to SGD). Potential cancellations are taken care of for sufficiently small $\eta > 0$ and $\rho > 0$ by the argument above. Now collecting all constraints that are already stated in the Tensor Program formulation at the respective step concludes the proof.

IV.3.2. PROOF OF THEOREM 17

A stable *bcd*-parameterization is nontrivial if and only if $\mathring{f}_t = \Theta(1)$ if and only if $\mathring{\theta}'_{L+1} = 1$ or $\mathring{\theta}'_{L\nabla} = 1$.

IV.3.3. PROOF OF THEOREM 18

A stable *bcd*-parametrization is feature learning in layer l if and only if the feature update scaling $\mathring{\theta}_l = 1$ where

$$\theta_l = n^{-r_l}, \quad r_l := \min(b_{L+1}, c_{L+1}, d_{L+1} + 1/2) + \min_{m=1}^{l}(c_m - \mathbb{I}(m \neq 1)).$$

Hence a stable *bcd*-parametrization is feature learning in layer l if and only if $r_l = 0$.

Since for all $l_1 \leq l_2$, it holds that $r_{l_1} \geq r_{l_2} \geq 0$, we get the equivalence for any $l_0 \in [L]$: A stable *bcd*-parametrization is feature learning in layer l_0 if and only if it is feature learning in layer l for all $l \geq l_0$ if and only if $r_{l_0} = 0$.

IV.3.4. PROOF OF THEOREM 20

Given a stable *bcd*-parametrization, perturbation triviality is fulfilled if and only if $\mathring{\tilde{\theta}}'_{L+1} = 0$ and $\mathring{\tilde{\theta}}'_{L\nabla} = 0$, where $\tilde{\theta}'_{L+1} = n^{1/2-d_{L+1}}$ and $\tilde{\theta}'_{L\nabla} = n\theta_\nabla \tilde{\theta}_L = n^{1-\min(b_{L+1},c_{L+1})-\tilde{r}}$, hence if and only if $d_{L+1} > 1/2$ and $\min(b_{L+1}, c_{L+1}) + \tilde{r} > 1$.

In that case, $\mathring{\tilde{f}}_t = \mathring{f}_t$, but $\mathring{f}_t$ may still be affected by non-vanishing SAM perturbations in δW_t^{L+1} and δx_t^L. Only when all SAM perturbations vanish are we effectively only using SGD. By definition, the perturbation scale in the l-th layer vanishes if and only if $\mathring{\tilde{\theta}}_l = 0$, where $\tilde{\theta}_l = n^{-\tilde{r}_l}$ with $\tilde{r}_l = \min(b_{L+1}, c_{L+1}) + 1/2 + \min_{m=1}^{l}(d_m - \mathbb{I}(m \neq 1))$, hence if and only if $\tilde{r}_l > 0$. Since $\tilde{r}_l \geq \tilde{r}_L = \tilde{r}$ for all $l \leq L$, we get $\mathring{\tilde{\theta}}_l = 0$ for all $l \in [L]$ if and only if $\tilde{r} > 0$. Similarly, for any reference layer $l_0 \in [L]$, we get $\mathring{\tilde{\theta}}_l = 0$ for all $l \leq l_0$ if and only if $\tilde{r}_{l_0} > 0$. In words, for any $l_0 \in [L]$, we have vanishing perturbations in layer l_0 if and only if we have vanishing perturbations until layer l_0 if and only if $\tilde{r}_{l_0} > 0$.

Altogether, a stable *bcd*-parametrization has vanishing perturbations if and only if $\tilde{r} > 0$, $d_{L+1} > 1/2$ and $\min(b_{L+1}, c_{L+1}) + \tilde{r} > 1$. This case reduces to the results in Yang and Hu [52] in the limit. Since stability requires $c_{L+1} \geq 1$ and $\tilde{r} \geq 0$, we can rewrite the equivalence conditions as $d_{L+1} \geq 1/2$ and $\tilde{r} > \max(0, 1 - b_{L+1})$.

IV.3.5. PROOF OF THEOREM 22

Recall $\tilde{\theta}_{W^1} := n^{-(d+d_1)}\theta_\nabla$, $\tilde{\theta}_{W^l} := n^{1-(d+d_l)}\theta_\nabla$ and, for the last layer $\tilde{\theta}_{W^{L+1}} := n^{-(d+d_{L+1})}$.

As opposed to perturbation nontriviality, we are not only interested in $\tilde{\theta}_l = \max(\tilde{\theta}_{l-1}, \tilde{\theta}_{W^l}) = \max_{m=1}^{l} \tilde{\theta}_{W^m} \to 1$, but in a non-vanishing contribution of the perturbations in layer l, i.e. $\mathring{\tilde{\theta}}_{W^l} = 1$ or, for the last layer, $\mathring{\tilde{\theta}}_{L+1} = 1$.

IV.3.6. PROOF OF THEOREM 23

The limit of the gradient norm is defined as a NE⊗OR⊤ program scalar (V.1). Note that for $b_{L+1} > 1/2$, the last-layer scaling strictly dominates all other scalings leading to the simplified gradient norm formula.

Now consider an arbitrary stable choice of layerwise initialization variances $\{b_l\}_{l\in[L+1]}$ and learning rates $\{c_l\}_{l\in[L+1]}$. To fulfill the gradient norm constraints (III.1), we have to choose $d_l = C = 1/2$ for all $l \in [L+1]$, because stability requires $\min(b_{L+1}, c_{L+1}) \geq 1/2$. Now stability of the output function perturbations requires $d \geq 1/2$, where $d > 1/2$ yields vanishing perturbations and $d = 1/2$ yields effective last-layer SAM through the term $\tilde{\delta} W_t^{L+1} \tilde{x}_t^L$. After choosing $d \geq 1/2$, we get $\tilde{r} \geq \min(b_{L+1}, c_{L+1}) \geq 1/2 > 0$ which implies vanishing perturbations in all hidden layers.

IV.3.7. PROOF OF PROPOSITION 24

To achieve non-vanishing gradient norm contribution of the last layer in (III.1), we need to choose $d_{L+1} = 1/2$, which requires $d \geq 1/2$ for stability of the output function perturbations. Achieving non-vanishing gradient norm contributions of all layers requires $d_1 = 1/2 - \min(b_{L+1}, c_{L+1})$ and

$d_l = 1 - \min(b_{L+1}, c_{L+1})$ for $l \in [2, L]$, which results in $\tilde{r} = d \geq 1/2 > 0$ which implies vanishing perturbations in all hidden layers.

IV.3.8. PROOF OF THEOREM 25

Given a stable bcd-parametrization, we know $d + d_{L+1} \geq 1$, so that the feature learning constraint r is not affected by any stable choice of $d \cup \{d_l\}_{l\in[L+1]}$. The maximal stable choice of layerwise initialization variances $\{b_l\}_{l\in[L+1]}$ and learning rates $\{c_l\}_{l\in[L+1]}$ that constitute μP is therefore unaffected by the perturbation scalings $d \cup \{d_l\}_{l\in[L+1]}$.

Stability of the output function perturbations requires $b_{L+1} + \tilde{r} \geq 1$. Hence if $b_{L+1} < 1$, then $\tilde{r} \geq 1 - b_{L+1} > 0$, which implies vanishing perturbations in all hidden layers.

From now on consider $b_{L+1} \geq 1$. Recall $c_\nabla := \min(b_{L+1}, c_{L+1})$. In μP, $c_\nabla = 1$, but effective perturbations in all layers can be achieved more generally for $c_\nabla \geq 1$. Choosing $d_1 = 1/2 - c_\nabla$ saturates the gradient norm constraint (III.1). To reach effective perturbations already in the first layer $\tilde{r}_1 = c_\nabla + d + d_1 = 0$, we need $d = -1/2$. For perturbation stability and last-layer effective perturbations, we need $d + d_{L+1} = 1$ which requires $d_{L+1} = 3/2$. Achieving perturbation stability and effective perturbations in all hidden layers requires $\tilde{\theta}_{W^l} = 1$ which is equivalent to $c_\nabla + d + d_l - \mathbb{I}(l \neq 1) = 0$. For $l \in [2, L]$, we therefore need $d_l = 3/2 - c_\nabla$. This choice of $\{d_l\}_{l\in[L+1]}$ achieves effective perturbations in all layers.

To show uniqueness we iterate through all possibilities of saturating the norm bound constraint (III.1). We have considered the cases $d_{L+1} = 1/2$ in (b) leading to vanishing perturbations in all hidden layers and $d_1 = 1/2 - c_\nabla$ in (c) with only one choice for effective perturbations in all layers. Lastly consider $d_l = 1 - c_\nabla$ for $l \in [2, L]$ for non-vanishing gradient contribution of the hidden layers. Note that all hidden layers play the same role in all relevant constraints. Effective perturbations in any hidden layer $l \in [2, L]$ requires $\tilde{\theta}_{W^l} = 1$ for which we need $d = 0$. But then, as $d_1 \geq 1/2 - c_\nabla$, it holds that $\tilde{r}_1 \geq 1/2$ implying vanishing perturbations in the first layer. This shows the uniqueness of (2).

For the gradient norm statements, note that the gradient norm $\|v_t\|$ can be written as a NE$\otimes$OR$\top$computation rule (IV.2) where the layer scalings in this parameterization are $\Theta(1)$ for the input layer, $\Theta(n^{-1/2})$ for hidden layers and $\Theta(n^{-1})$ for the output layer. Now the Tensor Program master theorem immediately implies the result.

IV.3.9. PROOF OF PROPOSITION 26

Perturbation nontriviality with respect to any hidden layer is equivalent to $\tilde{r} = 0$. Since $\min(b_{L+1}, c_{L+1}) \leq 1$, we get $\min(b_{L+1}, c_{L+1}) + \tilde{r} \leq 1$. Since stability requires $\min(b_{L+1}, c_{L+1}) + \tilde{r} \geq 1$, we get $\min(b_{L+1}, c_{L+1}) + \tilde{r} = 1$, which implies perturbation nontriviality with respect to the output.

IV.3.10. PROOF OF PROPOSITION 27

The constraint is the same constraint as in Theorem 22, which implies effective perturbations in the first layer. Now $\tilde{r}_l \leq \tilde{r}_1 = 0$ implies perturbation nontriviality in all hidden layers due to Theorem 20.

IV.4. Analytic expression of the features after first SAM update

Below we state the analytic expression of the first SAM update, but leave a closer analysis of its fine-grained dynamics in comparison to SGD to future work. Before looking into the effective perturbation regime, we restate Lemma H.37 in Yang and Hu [52] with a more detailed proof.

First, we define $\ell \in [L]$ as the unique index that satisfies $\theta_L = \cdots = \theta_\ell = 1 > \theta_{\ell-1} \geq \cdots \geq \theta_1$. In words, ℓ is the first layer in which feature learning occurs. Analogously, we define $\tilde{\ell} \in [L]$ as the unique index that satisfies $1 = \frac{\tilde{\theta}_L}{\tilde{\theta}_L} = \cdots = \frac{\tilde{\theta}_{\tilde{\ell}}}{\tilde{\theta}_L} > \frac{\tilde{\theta}_{\tilde{\ell}-1}}{\tilde{\theta}_L} \geq \cdots \geq \frac{\tilde{\theta}_1}{\tilde{\theta}_L}$.

Lemma 31 (Features after first SGD step) *Defining $Z_t^l := Z^{h_t^l}$, $\gamma^l(\eta) = \mathbb{E}\phi(Z_0^l)\phi(Z_1^l)$ for $l \geq 1$, $\gamma^0 = \xi_0^T \xi$ and $\gamma_{11}^l(\eta) = \mathbb{E}\phi'(Z_0^l)\phi'(Z_1^l)$, we have*

$$Z_1^{\ell-1} = Z_0^{\ell-1}, \ldots, Z_1^1 = Z_0^1,$$

and, for all $l \geq \ell$,

$$Z_1^l = Z_0^l + \mathbb{I}_{l>\ell} \hat{Z}^{W_0^l \delta x_1^{l-1}} + \eta \beta^l Z^{dx_0^l} \phi'(Z_0^l),$$

where β^l is defined recursively by

$$\beta^l = \beta^l(\eta) = -\mathring{\chi}_0 \gamma^{l-1}(\eta) + \beta^{l-1}(\eta)\gamma_{11}^{l-1}(\eta),$$

with $\beta^{\ell-1} = 0$. Note that $\beta^l(0) < 0$ for all $l \geq \ell$.

Proof By the defining infinite-width equations, assuming $\mathring{\theta}_{W^l/l} = 1$ (so minimal stable choice of c_l),

$$Z_1^l = Z_0^l + \mathring{\theta}_{(\ell-1)/\ell} Z^{W_0^l \delta x_1^{l-1}} - \eta \mathring{\chi}_0 \gamma^{l-1} Z^{dx_0^l} \phi'(Z_0^l).$$

At $l = \ell$, we get $\mathring{\theta}_{(\ell-1)/\ell} = 0$, whereas for $l > \ell$ we get $\mathring{\theta}_{(l-1)/l} = 1$, which results in $\mathring{\theta}_{(\ell-1)/\ell} = \mathbb{I}_{l>\ell}$.

Now, for $l > \ell$, the second term decomposes into $\hat{Z}^{W_0^l \delta x_1^{l-1}}$ and

$$\dot{Z}^{W_0^l \delta x_1^{l-1}} = Z^{dh_0^l} \mathbb{E} \frac{\partial Z^{\delta x_1^{l-1}}}{\partial \hat{Z}^{(W_0^l)^T dh_0^l}}.$$

Since by induction hypothesis,

$$Z^{\delta x_1^{l-1}} = \phi(Z_1^{l-1}) - \phi(Z_0^{l-1}) = \phi\left(Z_0^{l-1} + \mathbb{I}_{l>\ell} \hat{Z}^{W_0^l \delta x_1^{l-1}} + \eta \beta^{l-1} Z^{dx_0^{l-1}} \phi'(Z_0^{l-1})\right) - \phi(Z_0^{l-1}),$$

where $Z^{dx_0^{l-1}} = Z^{(W_0^l)^T dh_0^l}$ is the only dependence on $\hat{Z}^{(W_0^l)^T dh_0^l}$, we get

$$\frac{\partial Z^{\delta x_1^{l-1}}}{\partial \hat{Z}^{(W_0^l)^T dh_0^l}} = \phi'(Z_1^{l-1}) \eta \beta^{l-1} \phi'(Z_0^{l-1}).$$

Plugging the derivative back into the defining equation and noticing that $Z^{dh_0^l} = Z^{dx_0^l} \phi'(Z_0^l)$ concludes the proof. ■

An analogous analysis for the perturbation at initialization shows.

Lemma 32 (Feature perturbation at initialization) *The perturbation trivial layers fulfill*

$$Z^{\tilde{h}_0^{\tilde{\ell}-1}} = Z^{h_0^{\tilde{\ell}-1}}, \dots, Z^{\tilde{h}_0^1} = Z^{h_0^1},$$

and, for all $l \geq \tilde{\ell}$*,*

$$Z^{\tilde{h}_0^l} = Z^{h_0^l} + \mathbb{I}_{l>\tilde{\ell}} \hat{Z}^{W_0^l \tilde{\delta} x_0^{l-1}} + \rho \tilde{\beta}^l Z^{dx_0^l} \phi'(Z^{h_0^l}),$$

where $\tilde{\beta}^l$ *independent of* η *is defined recursively by*

$$\tilde{\beta}^l = \tilde{\beta}^l(\rho) = \frac{\mathring{\chi}_0}{\mathring{\|}\nabla L_0\|} \mathbb{E}[\phi(Z^{h_0^{l-1}})\phi(Z^{\tilde{h}_0^{l-1}})] + \tilde{\beta}^{l-1} \mathbb{E}[\phi'(Z^{h_0^{l-1}})\phi'(Z^{\tilde{h}_0^{l-1}})]$$

with $\tilde{\beta}^{\tilde{\ell}-1} = 0$*. Note that* $\tilde{\beta}^l(0) > 0$ *for all* $l \geq \tilde{\ell}$*.*

Remark 33 *If* $\tilde{\ell} = 1$*, in the definition of* $\tilde{\beta}^l$ *replace* $\mathbb{E}[\phi(Z^{h_0^{l-1}})\phi(Z^{\tilde{h}_0^{l-1}})]$ *by* $\xi_0^T \xi$*.*

Now we are ready to state the closed form expression for the first SAM update.

Lemma 34 (Features after first SAM update) *Defining* $Z_t^l := Z^{h_t^l}$ *and* $\tilde{Z}_t^l := Z^{\tilde{h}_t^l}$*, we have*

$$Z_1^{\ell-1} = Z_0^{\ell-1}, \dots, Z_1^1 = Z_0^1,$$

and, for all $l \geq \ell$*,*

$$Z_1^l = Z_0^l + \mathbb{I}_{l>\ell} \hat{Z}^{W_0^l \delta x_1^{l-1}} + \eta \beta^l Z^{dx_{SAM,0}^l} \phi'(\tilde{Z}_0^l) + \eta \gamma^l Z^{dh_0^l},$$

where β^l *is defined recursively by*

$$\beta^l = \beta^l(\eta) = -\mathring{\chi}_0 \mathbb{E}[\phi(\tilde{Z}_0^{l-1})\phi(Z_1^{l-1})] + \beta^{l-1}(\eta) \mathbb{E}[\phi'(Z_1^{l-1})\phi'(\tilde{Z}_0^{l-1})],$$

with $\beta^{\ell-1} = 0$*, and* $\gamma^l = \gamma^l(\eta)$ *is recursively defined by*

$$\gamma^l := \beta^{l-1} \rho \tilde{\beta}^{l-1} \mathbb{E}[\phi'(Z_1^{l-1})\phi'(Z_0^{l-1})\phi''(\tilde{Z}_0^{l-1}) Z^{dx_{SAM,0}^{l-1}}] + \gamma^{l-1} \mathbb{E}[\phi'(Z_0^{l-1})\phi'(Z_1^{l-1})],$$

with $\gamma^{\ell-1} = \gamma^\ell = 0$*.*

Remark 35 *If* $\ell = 1$*, in the definition of* β^l *replace* $\mathbb{E}[\phi(\tilde{Z}_0^{l-1})\phi(Z_1^{l-1})]$ *by* $(\xi_{t-1}^T \xi)$*.*

Proof By the defining infinite-width equations, for $l \geq \ell$, assuming $\mathring{\theta}_{W^l/l} = 1$ (so minimal stable choice of c_l),

$$Z_1^l = Z_0^l + \mathring{\theta}_{(l-1)/l} Z^{W_0^l \delta x_1^{l-1}} - \eta \mathring{\chi}_0 \mathbb{E}[\phi(\tilde{Z}_0^{l-1})\phi(Z_1^{l-1})] Z^{dx_{SAM,0}^l} \phi'(\tilde{Z}_0^l). \tag{IV.3}$$

At $l = \ell$, we get $\mathring{\theta}_{(\ell-1)/\ell} = 0$ and $\mathring{\theta}_{W^\ell/\ell} = 1$, whereas for $l > \ell$ we get $\mathring{\theta}_{(l-1)/l} = 1$ and $\mathring{\theta}_{W^l/l} = 1$ (under minimal stable choice of c_l), which results in $\mathring{\theta}_{(l-1)/l} = \mathbb{I}_{l>\ell}$. Now, for $l > \ell$, the second term decomposes into $\hat{Z}^{W_0^l \delta x_1^{l-1}}$ and $\dot{Z}^{W_0^l \delta x_1^{l-1}}$. For the rest of the proof it remains to analyse $\dot{Z}^{W_0^l \delta x_1^{l-1}}$.

Since by induction hypothesis,

$$\begin{aligned} Z^{\delta x_1^{l-1}} &= \phi(Z_1^{l-1}) - \phi(Z_0^{l-1}) \\ &= \phi\left(Z_0^{l-1} + \mathbb{I}_{l>\ell}\hat{Z}^{W_0^l \delta x_1^{l-1}} + \eta\beta^{l-1} Z^{dx_{SAM,0}^{l-1}}\phi'(\tilde{Z}_0^{l-1}) + \eta\gamma^{l-1} Z^{dh_0^{l-1}}\right) - \phi(Z_0^{l-1}), \end{aligned}$$

where $Z^{dx_{SAM,0}^{l-1}} = Z^{(W_0^l)^T dh_{SAM,0}^l} + \rho\mathring{\tilde{\theta}}_{W^l}\frac{\mathring{\chi}_0}{\|\mathring{\nabla} L_0\|}\mathbb{E}[Z^{dh_0^l} Z^{dh_{SAM,0}^l}] Z^{x_0^{l-1}}$ with the second term independent of $(W_0^l)^T$ and by Lemma 32 we know $\tilde{Z}_0^{l-1} = Z_0^{l-1} + \mathbb{I}_{l-1>\tilde{\ell}}\hat{Z}^{W_0^{l-1}\tilde{\delta} x_0^{l-2}} + \rho\tilde{\beta}^{l-1} Z^{dx_0^{l-1}}\phi'(Z_0^{l-1})$, where only the last term with $Z^{dx_0^{l-1}} = Z^{(W_0^l)^T dh_0^l}$ influences $\dot{Z}^{W_0^l \delta x_1^{l-1}}$, we get

$$\dot{Z}^{W_0^l \delta x_1^{l-1}} = Z^{dh_0^l}\mathbb{E}\frac{\partial Z^{\delta x_1^{l-1}}}{\partial \hat{Z}^{(W_0^l)^T dh_0^l}} + Z^{dh_{SAM,0}^l}\mathbb{E}\frac{\partial Z^{\delta x_1^{l-1}}}{\partial \hat{Z}^{(W_0^l)^T dh_{SAM,0}^l}}, \tag{IV.4}$$

with

$$\frac{\partial Z^{\delta x_1^{l-1}}}{\partial \hat{Z}^{(W_0^l)^T dh_{SAM,0}^l}} = \phi'(Z_1^{l-1})\eta\beta^{l-1}\phi'(\tilde{Z}_0^{l-1}),$$

and, using $Z^{dh_0^{l-1}} = Z^{dx_0^{l-1}}\phi'(Z_0^{l-1}) = Z^{(W_0^l)^T dh_0^l}\phi'(Z_0^{l-1})$, yields

$$\begin{aligned} \frac{\partial Z^{\delta x_1^{l-1}}}{\partial \hat{Z}^{(W_0^l)^T dh_0^l}} &= \phi'(Z_1^{l-1})\left(\eta\beta^{l-1} Z^{dx_{SAM,0}^{l-1}}\phi''(\tilde{Z}_0^{l-1})\frac{\partial \tilde{Z}_0^{l-1}}{\partial \hat{Z}^{(W_0^l)^T dh_0^l}} + \eta\gamma^{l-1}\phi'(Z_0^{l-1})\right) \\ &= \phi'(Z_1^{l-1})\left(\eta\beta^{l-1} Z^{dx_{SAM,0}^{l-1}}\phi''(\tilde{Z}_0^{l-1})\rho\tilde{\beta}^{l-1}\phi'(Z_0^{l-1}) + \eta\gamma^{l-1}\phi'(Z_0^{l-1})\right). \end{aligned}$$

Plugging Eq. (IV.4) back into the defining equation (IV.3) and noticing that $Z^{dh_{SAM,0}^l} = Z^{dx_{SAM,0}^l}\phi'(\tilde{Z}_0^l)$ as well as $Z^{dh_0^l} = Z^{dx_0^l}\phi'(Z_0^l)$ concludes the proof. ■

V. Additional theoretical considerations

V.1. Alternative bcd-definition

A desirable definition of parametrizations for the SAM update rule should certainly allow perturbations that have a stable but non-vanishing effect on the perturbed (pre-)activations and perturbed output function. Otherwise we would not need to perturb the inactive layers in the first place. Writing out the gradient norm, it turns out that for $b_{L+1} > 1/2$ (such as in μP) only the last layer dominates the gradient norm, yielding $\|\nabla_w L(f(\xi_t; W_t), y_t))\| \approx L'(f_t(\xi_t), y_t)\|x^L\| = \Theta(n^{1/2})$ in the limit. Both for achieving a non-vanishing effect in every layer but also if one wants to balance the contributions of each layer to the gradient norm, the gradient has to be scaled layerwise such as $\beta \odot \nabla_w L$ with $\beta^l = n^{-d_l}$, which changes the direction away from the original gradient direction. In this way, our definition allows maximal stable perturbations in each layer while using a single perturbation. The choice of d_l then determines whether the original gradient direction should be retained with $\beta^i = \beta^j$ for all $i, j \in [L+1]$, whether the norm contributions should be balanced or whether another layer weighting is preferred, for example to achieve maximal stable perturbations in each layer.

If we want to take a step in the original gradient direction, but adapt the perturbation scaling in each layer to the maximal stable scaling at the same time, we need to compute a layerwise perturbation that may require multiple outer backward passes and use an update rule such as

$$W_t^l = W_t^l - \eta n^{-c_l} \nabla_{W^l} L \left(f\left(\xi_t; W_t + \rho n^{-d_l} \frac{\nabla_w L}{\|\nabla_w L\|} \right), y_t \right).$$

This update rule can be computationally much more expensive as it requires to evaluate the outer gradient on multiple perturbed weights at each time step. It is therefore arguably less interesting for practitioners. Since the perturbation scaling only comes in after the dh, dx iteration and always $\|\nabla_w L(f(\xi_t; W_t), y_t))\| = \Theta(n^{1/2})$, the scalings for this update rule are covered by the above analysis of our bcd-definition when setting $d = 1/2$ to imitate the gradient norm scaling, and replacing the normalized TP scalar $\|v_t\|$ with the normalized TP scalar

$$\|\bar{\nabla L}_t\| := \chi_t \left(\theta_\nabla^2 \frac{(dh_t^1)^T dh_t^1}{n} (\xi_t^T \xi_t) + \sum_{l=2}^{L} n\theta_\nabla^2 \frac{(dh_t^l)^T dh_t^l}{n} \frac{(x_t^{l-1})^T x_t^{l-1}}{n} + \frac{(x_t^L)^T x_t^L}{n} \right)^{1/2}.$$

For all $t \geq 0$, the limit of this normalized gradient norm is given by

$$\|\mathring{\bar{\nabla}} L_t\| = \mathring{\chi}_t \left(\mathring{\theta}_\nabla^2 \mathbb{E}[Z^{(dh_t^1)^2}](\xi_t^T \xi_t) + \sum_{l=2}^{L} \mathring{\theta}_\nabla^{2\prime} \mathbb{E}[Z^{(dh_t^l)^2}] \mathbb{E}[Z^{(x_t^{l-1})^2}] + \mathbb{E}[Z^{(x_t^L)^2}] \right)^{1/2}, \quad \text{(V.1)}$$

where the first term vanishes as $\mathring{\theta}_\nabla = 0$, $\mathring{\chi}_t = L'(\mathring{f}_t(\xi_t), y_t)$ and $\theta_\nabla^{2\prime} := n\theta_\nabla^2$ where $\mathring{\theta}_\nabla^{2\prime} = 0$ if and only if $b_{L+1} > 1/2$ and $\mathring{\theta}_\nabla^{2\prime} = 1$ if and only if $b_{L+1} = 1/2$ (since $c_{L+1} \geq 1$ for stability of the output function). Consequently, the updated stability, nontriviality and perturbation nontriviality constraints remain the same with the choice $d = 1/2$.

The maximal update parametrization with maximal perturbations again remains invariant in layerwise initialization variances b_l and learning rates c_l but uniquely becomes $d_l = \begin{cases} -3/2 & l = 1, \\ -1/2 & l \in [2, L], \\ 1/2 & l = L+1. \end{cases}$ without equivalent scalings because $\|\nabla_w L_t\| = \Theta(n^{1/2})$.

Another potential definition could decouple the scalings in the numerator and the denominator, and simply scale all layerwise gradient norms in the denominator to be width-indepdendent $\Theta(1)$. In this way all layers contribute to the gradient norm by design and we recover $\Theta(1)$ gradient norm scaling, however we lose the control over the norm of the perturbation of all parameters potentially reducing it beyond ρn^{-d}. Our ablation studies find no significant performance differences between this alternative definition and our choice.

V.2. The criterion $\varepsilon_t^l = \Theta(\delta W_t^l)$ for effective perturbations in μP

As in Yang et al. [55], we might be interested in a simple criterion to check for achieving effective perturbations, except quantifying the correct scaling by writing out the NE⊗OR⊤ program. To achieve effective perturbations in μP, there is a simple criterion to fulfill: $\tilde{\delta} W^l = \Theta(\delta W^l)$. In short, this is because the perturbation follows a similar direction as the update, both correlated with the

incoming activations (inducing LLN-like scaling behaviour), so that if the update $\delta W_t^l x_t^{l-1}$ and the perturbation $\tilde{\delta} W_t^l \tilde{x}_t^{l-1}$ shall have the same scaling, then it will generally hold that $\tilde{\delta} W^l = \Theta(\delta W^l)$.

In μP, we have $\delta W_t^l x_t^{l-1} = \Theta(1)$, so that $\tilde{\delta} W^l = \Theta(\delta W^l)$ provides a criterion to achieve effective perturbations $\tilde{\delta} W_t^l \tilde{x}_t^{l-1} = \delta W_t^l x_t^{l-1} = \Theta(1)$. If not in μP, we do not have $\delta W_t^l x_t^{l-1} = \Theta(1)$, so that we might have to choose $\tilde{\delta} W_t^l$ larger than δW_t^l to achieve effective perturbations. However $W_t^l x_t^{l-1} = o(1)$ is undesirable, as the l-th layer then does not learn features in the infinite-width limit.

Below we explicitly write out the scalings of updates δW^l and perturbations $\tilde{\delta} W^l$ for each layer type. The interested reader may take this as an opportunity to familiarize themselves with these scaling arguments.

In the last layer $l = L + 1$, observe $W_0^{L+1} = \Theta(n^{-b_{L+1}})$ and $\delta W_t^{L+1} = -\eta n^{c_{L+1}} \tilde{\chi}_t \|\tilde{x}_t^L\| = \Theta(n^{-c_{L+1}})$. Thus $\tilde{\delta} W_t^{L+1} = \Theta(W_t^{L+1})$, $t > 0$, if and only if $d + d_{L+1} = \min(b_{L+1}, c_{L+1})$. Hence, in μP we indeed get $\tilde{\delta} W_t^{L+1} = \Theta(W_t^{L+1}) = \Theta(n^{-1})$ for all $t \geq 0$.

In the first layer $l = 1$, $W_0^1 = \Theta(n^{-b_1})$ and

$$\delta W_t^1 = -\eta n^{-c_1} \tilde{\chi}_t d h_{SAM,t}^1 \xi_t^T = \Theta(n^{-c_1 - \min(b_{L+1}, c_{L+1}, d + d_{L+1})}).$$

The perturbation simply replaces the scaled learning rate ηn^{-c_1} by the scaled perturbation radius $\rho n^{-(d+d_1)}$ and acts on the unperturbed gradient $\tilde{\delta} W_t^1 = \rho n^{-(d+d_1)} \chi_t d h_t^1 \xi_t^T$, so that $\tilde{\delta} W_t^1 = \Theta(W_t^1)$, $t > 0$, if and only if $d + d_1 + \min(b_{L+1}, c_{L+1}) = \min(b_1, c_1 + \min(b_{L+1}, c_{L+1}, d + d_{L+1}))$. For μP, this results in $\tilde{\delta} W_t^1 = \Theta(W_t^1) = \Theta(1)$ for all $t \geq 0$.

In the hidden layers $l \in [2, L]$, we have initialized weights $W_0^l = \Theta(n^{-b_l})$, weight updates $\delta W_t^l = -\eta n^{-c_l} \tilde{\chi}_t d h_{SAM,t}^l (\tilde{x}_t^{l-1})^T = \Theta(n^{-c_l - \min(b_{L+1}, c_{L+1}, d + d_{L+1})})$, and the weight perturbation $\tilde{\delta} W_t^l = \rho n^{-(d+d_l)} \chi_t d h_t^l (x_t^{l-1})^T = \Theta(n^{-(d+d_l) - \min(b_{L+1}, c_{L+1})})$, so that $\tilde{\delta} W_t^l = \Theta(W_t^l)$, $t > 0$, if and only if $d + d_l + \min(b_{L+1}, c_{L+1}) = \min(b_l, c_l + \min(b_{L+1}, c_{L+1}, d + d_{L+1}))$.

In μP, note that in hidden layers $l \in [2, L]$ initial weights behave as CLT requiring $W_0^l = \Theta(n^{-1/2})$, while updates $\delta W_t^l = \Theta(n^{-1})$ behave as LLN since they are highly correlated with the incoming activations x_t^{l-1}. As perturbations also follow the gradient direction they are also highly correlated with x_t^{l-1}. For $t > 0$, the perturbation radius $\rho n^{-(d+d_l)}$ replaces learning rate ηn^{-c_l} as in the other layers so that the perturbation always has the same scaling as the weight updates $\tilde{\delta} W_t^l = \Theta(\delta W_t^l) = \Theta(n^{-1})$, but note that the total scaling of the hidden layer weights is therefore larger than the perturbation scaling, $\tilde{\delta} W_t^l = o(W_t^l) = o(n^{-1/2})$.

V.3. Overview over choices of d_l and d

Since for some combinations of architectures and datasets it turns out that performing SAM on a subset of layers performs better than effective perturbations in all layers [35], we would like to know how to choose d and d_l to adjust which layers should be effectively perturbed and which should have vanishing weight perturbations. In practice, simply set all perturbations that should vanish to 0 by design, and use the global scaling d and relative scalings d_l from μP^2 for the perturbed layers. This section is instead interested in a complete characterization of all possible choices of $\{d_l\}_{l \in [L+1]}$ and d. The heuristic derivation only requires the gradient norm constraints (III.1) and the perturbation

	Effective perturbations possible			Gradient norm may be dominated by		
	input-like	hidden-like	output-like	input-like	hidden-like	output-like
$d = -1/2$	✓	✓	✓	✓	✗	✗
$d \in (-1/2, 0)$	✗	✓	✓	✓	✗	✗
$d = 0$	✗	✓	✓	✓	✓	✗
$d \in (0, 1/2)$	✗	✗	✓	✓	✓	✗
$d = 1/2$	✗	✗	✓	✓	✓	✓
$d > 1/2$	✗	✗	✗	✓	✓	✓

Table V.1: **(Characterization of perturbation scalings)** Overview over the regimes of all possible choices of d and d_l. A layer is effectively perturbed if and only d_l satisfies (V.2). At least one layer must satisfy equality in its gradient norm constraint (III.1). This table summarizes which layers can exhibit effective perturbations, and which may dominate the gradient norm, given a choice of d. The choice $d < -1/2$ results in perturbation blowup $\tilde{r} < 0$. At the critical $d = -1/2$ (respectively, $d = 0$; $d = 1/2$) a input-like (respectively hidden-like; output-like) layer is effectively perturbed if and only if it dominates the gradient norm. Consequently $d = -1/2$ implies effective perturbations in at least one input-like layer.

stability constraints that require $\tilde{\delta}W^1 = O(1)$ and $\tilde{\delta}W^l = O(n^{-1})$ for $l > 1$ given by

$$d_l \geq \begin{cases} -c_\nabla - d & \text{if } l \text{ is input-like,} \\ 1 - c_\nabla - d & \text{if } l \text{ is hidden-like,} \\ 1 - d & \text{if } l \text{ is output-like,} \end{cases} \tag{V.2}$$

where a layer is effectively perturbed if and only if equality holds in the respective perturbation stability inequality. This heuristic claim yields the characterization of all phases of the choices of perturbation scalings d and d_l in Table V.1 and allows us to formulate a simple rule of how to choose d and d_l given the information which layers should be effectively perturbed, and which should have vanishing weight perturbations.

Choice of perturbation scaling from list of layers to effectively perturb. We denote the set of all layers by $\mathcal{L}$, whereas the subset of layers, which we want to effectively perturb, is denoted by $\mathcal{L}_{SAM} \subseteq \mathcal{L}$.

1. If there exists an input-like layer $l \in \mathcal{L}_{SAM}$, set $d = -1/2$. Input-like layers are effectively perturbed if and only if $d_l = 1/2 - c_\nabla$. Hidden-like (respectively, output-like) layers are effectively perturbed if and only if $d_l = 3/2 - c_\nabla$ (respectively, $d_l = 3/2$). For all layers that have vanishing weight perturbations, do not perturb these weights or choose $d_l > 1/2 - c_\nabla$ for input-like, $d_l > 3/2 - c_\nabla$ for hidden-like and $d_l > 3/2$ for output-like layers.
2. If all input-like layers should have vanishing weight perturbations but there exists a hidden-like layer $l \in \mathcal{L}_{SAM}$, set $d = 0$. Hidden-like layers are effectively perturbed if and only if $d_l = 1 - c_\nabla$. Output-like layers are effectively perturbed if and only if $d_{L+1} = 1$. For all layers that have vanishing weight perturbations, do not perturb these weights, or set $d_l > c_\nabla$ for input-like, $d_l > 1 - c_\nabla$ for hidden-like and $d_l > 1$ for output-like layers (as required by the perturbation stability and gradient norm constraints).

3. If both all input-like and all hidden-like layers have vanishing weight perturbations, but there exists some output-like layer $l \in \mathcal{L}_{SAM}$, then set $d = 1/2$. Output-like layers are effectively perturbed if and only if $d_l = 1/2$. For all layers that have vanishing weight perturbations, do not perturb these weights or set $d_l \geq 1/2 - c_\nabla$ for input-like, $d_l \geq 1 - c_\nabla$ for hidden-like and $d_l > 1/2$ for output-like layers (as required by the perturbation stability and gradient norm constraints).
4. If $\mathcal{L}_{SAM} = \emptyset$, then set $d > 1/2$ or simply perform SGD.

Example 1 (First-layer-only effective perturbations) *Instead of simply using the rule set above, we derive the necessary choice of perturbation scaling from the scaling equalities and the norm constraints (III.1). To achieve first-layer effective perturbations, but trivial weight perturbations in all other layers, we need $\tilde{\theta}_{W^1} = 1$ and $\mathring{\tilde{\theta}}_{W^l} = 0$, for which we will choose $\tilde{\theta}_{W^l} = n^{-1}$. This requires setting*

$$d_1 = -(c_\nabla + d), \qquad d_l = 2 - c_\nabla - d, \qquad d_{L+1} = 2 - d,$$

where one of the constraints (III.1) has to be fulfilled. Plugging the above d_l-choices into (III.1) results in the constraints $d \leq -1/2$, $d \leq 1$, $d \leq 3/2$, hence choose $d = -1/2$ so that only the first layer contributes non-vanishingly to the gradient norm. Note that $\tilde{r} = 0$ and output perturbation nontriviality holds if and only if $\min(b_{L+1}, c_{L+1}) = 1$ (as in μP). We apply this perturbation scaling in Appendix VII.2 to show that propagating perturbations from early layers are not enough to inherit SAM's inductive bias that leads to improved generalization performance.

V.4. Extension to SAM without gradient normalization

Andriushchenko and Flammarion [1] and Andriushchenko et al. [2] consider the SAM update without normalizing the gradient in the adversarial ascent. The corresponding update rule is given by

$$W_t = W_t - \eta \nabla_w L(f(\xi_t; W_t + \rho v_t, y_t)), y_t), \qquad v_t = \nabla_w L(f(\xi_t; W_t).$$

The NE⊗OR⊤ program for this update rule with arbitrary $v_t^l = n^{-c_l} \nabla_w L(f(\xi_t; W_t)$ is also easily adapted from the above derivation. Just note that the gradient norm appears in an equation if and only if the perturbation radius ρn^{-d} appears. Without dividing by $\|v_t\|$, the parameter d becomes superfluous. Simply set $d = 0$ and remove the gradient norm constraints (III.1) to arrive at the NE⊗OR⊤ program and bcd-constraints for the update rule without gradient normalization.

Following the arguments in Appendix V.2, d_l plays a similar role as c_l as both scale similar gradients. We get effective perturbations in the l-th layer from the equation $d_l + \min(b_{L+1}, c_{L+1}) = c_l + \min(b_{L+1}, c_{L+1}, d_{L+1})$ in μP, which yields $d_l = c_l$ for all $l \in [L]$ (since $d_{L+1} = 1$ for stability). **In particular, in μP, the correct layerwise perturbation scaling of unnormalized gradients is given by the rule $\frac{\text{fan out}}{\text{fan in}}$ or the squared weight (update) spectral norm $\|W^l\|_*^2$** [55], which could be efficiently approximately tracked with a running power iteration.

Note that Dai et al. [10] argue that the normalizing the gradients for the perturbation is crucial (in standard parametrization) due to a stabilizing effect and an enhanced drift along manifolds of minima. Monzio Compagnoni et al. [34] find that unnormalized SAM gets stuck around saddles while SAM slowly escapes through additional Hessian-induced noise. This suggests that the additional effort of analysing the original SAM update rule with gradient normalization is necessary for practically

useful theory. From this paper's point of view, the gradient normalization may be adding stability via the $n^{-1/2}$ contribution which allows to scale down ρ less aggressively in practice.

V.5. Extension to Adaptive SAM

Adaptive SAM (ASAM) [28] is motivated by a sharpness definition that is invariant to parameter rescaling operators that leave the output function invariant, and can provide a further improvement over SAM of 0.5% to 1%, depending on the considered vision dataset and model [35]. Here we consider the two examples of elementwise rescaling operators (with $p = 2$) and layerwise rescaling operators (with $p = 2$), which are the best performing SAM variant in most settings in Müller et al. [35].

Proposition 36 *Neither elementwise ASAM, which performs* (SAM) *but using the perturbation rule* (V.5)*, nor layerwise ASAM, which performs* (SAM) *but using the perturbation rule* (V.7)*, can be written as a* NE⊗OR⊤*program.*

Proof sketch. Elementwise ASAM requires an elementwise multiplication of matrices, and layerwise ASAM requires calculating the Frobenius norm of a matrix. A NonLin operation only takes vectors as arguments, so NE⊗OR⊤calculations with a matrix require its multiplication with a vector. But then a single coordinate of the resulting vector contains a mixture of an entire row of that matrix. Since we are only allowed to define random vectors and matrices, this mixture cannot be disentangled by choosing a structured vector. ■

Although ASAM is not formally covered by our theory, we still expect that the ASAM perturbations are correlated with the gradient and therefore with the incoming activations, so that heuristically we can still expect LLN-like behaviour and apply our scaling condition. If the perturbation rules still behave LLN-like, then Table V.2 summarizes which layers are effectively perturbed under global scaling and provides the unique maximal perturbation scalings for all considered SAM variants. The correct perturbation scaling in μP for other perturbation rules that behave LLN-like can always be derived following the same steps:

1. In μP, it always holds that

$$W^l = \begin{cases} \Theta(1) & l = 1, \\ \Theta(n^{-1/2}) & l \in [2, L], \\ \Theta(n^{-1}) & l = L + 1, \end{cases} \quad \text{and} \quad \nabla_{W^l} L = \begin{cases} \Theta(\theta_\nabla) = \Theta(n^{-1}) & l \leq L, \\ \Theta(1) & l = L + 1. \end{cases} \tag{V.3}$$

2. Assuming the normalization term in the denominator is scaled to $\Theta(1)$, track the layerwise scalings of the numerator. Maximal stable perturbations are always achieved with

$$\tilde{\delta} W_t^l = \begin{cases} \Theta(1) & l = 1, \\ \Theta(n^{-1}) & l > 1. \end{cases} \tag{V.4}$$

   This yields constraints for achieving maximal stable perturbations in each layer.

	Perturbed under global scaling?			**For effective perturbations with μP^2:**			
	Input, biases, norm.	Other hidden layers	Output layer	Global ρ	Input-like	Hidden-like	Output-like
SAM	✗	✗	✓	$n^{1/2}$	$n^{1/2}$	$n^{-1/2}$	$n^{-3/2}$
Layer. ASAM	✗	✓	✗	1	1	n^{-1}	1
Elem. ASAM	✓	✓	✓	$n^{1/2}$	1	1	1
SAM-ON	✓	-	-	$n^{1/2}$	1	-	-

Table V.2: **(Layerwise perturbation scaling for effective perturbations in μP)** Without layerwise perturbation scaling (*left*), each SAM variant perturbs a different subset of layers at large width $n \to \infty$, but we provide the unique layerwise perturbation rescaling μP^2 (*right*) that achieves effective perturbations in all layers. This parameterization achieves hyperparameter transfer across widths.

3. Now replace the norm constraints (III.1) by tracking the scalings of each layer's contribution to the update rule's total normalization term.
4. To ensure normalization term scaling $\Theta(1)$, iterate through the layers l:
   (a) choose d_l to satisfy its norm constraint,
   (b) choose d to induce maximal stable perturbations in that layer,
   (c) choose all other $d_{l'}$, $l' \neq l$, minimal to both satisfy its norm constraint as well as perturbation stability $\tilde{\delta}W_t^l = \begin{cases} O(1) & l = 1, \\ O(n^{-1}) & l > 1. \end{cases}$
5. From the above configurations, choose the unique one that yields maximal stable perturbations in all layers.

If ASAM behaves Tensor Program-like, Table V.2 summarizes the subset of layers that are perturbed under global perturbation scaling, and how to achieve μP^2 with each SAM variant.

The experiments in Müller et al. [35] suggest that the role of normalization layer perturbations is particularly important. According to Table V.2, only SAM-ON and elementwise ASAM effectively perturb input-like layers under global scaling in μP. From a scaling perspective, normalization layers behave like input layers and therefore standard applications of SAM or layerwise ASAM do not effectively perturb them. Müller et al. [35, Section 5.3] make the same empirical observations in SP. Irrespective of the SAM variant, our scaling rules even allow to balance perturbations of different layer types and therefore provide precise understanding and control which layers should be perturbed across model scales.

V.5.1. ELEMENTWISE ASAM

If we want to be invariant to elementwise rescaling operators $T_w^l(x) = |W^l| \odot x$ where $x, W^l \in \mathbb{R}^{m \times n}$ and $\odot$ denotes elementwise multiplication, the resulting ASAM perturbation rule (where we

introduce (layer-wise) perturbation scalings $\{d\} \cup \{d_l\}_{l\in[L+1]}$) replaces (LP) and is given by

$$\tilde{\delta}W_t^l := \rho n^{-d} \frac{n^{-d_l}|W^l| \odot |W^l| \odot \nabla_{W^l} L(f(\xi_t; W_t), y_t)}{\|\nabla_{ASAM}^{elem}\|}, \tag{V.5}$$

with normalization

$$\|\nabla_{ASAM}^{elem}\| := \sum_{l=1}^{L+1} n^{-d_l} \left\| |W^l| \odot \nabla_{W^l} L(f(\xi_t; W_t), y_t) \right\|_F,$$

where the absolute values $|W^l|$ are computed and multiplied elementwise. To find the correct perturbation scalings, we track the typical elementwise scaling of each quantity as before.

Elementwise ASAM in μP. In μP, the layerwise weights and gradients scale as (V.3). For $\|\nabla_{ASAM}^{elem}\| = O(1)$, we therefore replace the constraints (III.1) by the constraints

$$d_l \geq 1/2 - c_\nabla, \text{ for } l \in [L], \qquad d_{L+1} \geq -1/2, \tag{V.6}$$

where we can choose $\{d_l\}_{l\in[L+1]}$ to achieve equality in at least one constraint to achieve $\|\nabla_{ASAM}^{elem}\| = \Theta(1)$.

The layerwise perturbations scale as $\tilde{\delta}W_t^l = n^{-d} \begin{cases} \Theta(n^{-d_1}\theta_\nabla) & l = 1, \\ \Theta(n^{-1-d_l}\theta_\nabla) & l \in [2, L], \\ \Theta(n^{-d_{L+1}}n^{-2}) & l = L+1. \end{cases}$

Stable and nontrivial perturbations in each layer are achieved under condition (V.4), which induces the constraints for optimal layerwise perturbation scaling

$$d + d_l = -c_\nabla, \text{ for } l \in [L], \qquad d + d_{L+1} = -1.$$

Irrespective which of the above norm constraints (V.6) we satisfy, we need $d = -1/2$ to achieve optimal layerwise perturbation scaling. Hence $d = d_{L+1} = -1/2$ and $d_l = 1/2 - c_\nabla$ for $l \in [L]$ is the unique choice of $\{d\} \cup \{d_l\}_{l\in[L+1]}$ modulo norm scaling equivalence that achieves $\Theta(1)$ perturbation scaling in all layers. With this choice all layers contribute non-vanishingly to the gradient norm. In μP $c_\nabla = 1$, so that $d_l = -1/2$ for all $l \in [L+1]$, so that ASAM does not require layerwise rescaling of the gradients, but upscaling of the perturbation by $n^{1/2}$ to achieve nontrivial perturbations in any layer. This may explain why ASAM often outperforms SAM in large models: By only requiring global scaling, ASAM achieves maximal stable perturbations in all layers if the perturbation radius is tuned globally at every width.

If instead of a global gradient norm $\|\nabla_{ASAM}^{elem}\|$, one would want to normalize in each layer separately with $\|\nabla_{ASAM}^{elem,l}\| := n^{-d_l}\||W^l| \odot \nabla_{W^l} L(f(\xi_t; W_t), y_t)\|_F$, the layerwise perturbation scalings become $\tilde{\delta}W_t^l = n^{-d} \begin{cases} \Theta(n^{-1/2}) & l = 1, \\ \Theta(n^{-3/2}) & l > 1. \end{cases}$ Again, to achieve maximal stable perturbations in all layers we need $d = -1/2$ and no layerwise adaptation of the gradient norm.

V.5.2. LAYERWISE ASAM

ASAM with layerwise rescaling as in Müller et al. [35] employs the layerwise transformations $T_w^l(x) = \|W^l\|_F \cdot x$. This ASAM perturbation rule replaces (LP) and is given by

$$\tilde{\delta} W_t^l := \rho n^{-d} \frac{n^{-d_l} \|W^l\|_F^2 \nabla_{W^l} L(f(\xi_t; W_t), y_t)}{\|\nabla_{ASAM}^{layer}\|}, \tag{V.7}$$

with normalization

$$\|\nabla_{ASAM}^{layer}\| := \sum_{l=1}^{L+1} n^{-d_l} \|W^l\|_F \|\nabla_{W^l} L(f(\xi_t; W_t), y_t)\|_F.$$

Layerwise ASAM in μP. In μP, we have $\|W^l\|_F = \begin{cases} \Theta(n^{1/2}) & l = 1, \\ \Theta(n^{1/2}) & l \in [2, L], \\ \Theta(n^{-1/2}) & l = L+1. \end{cases}$

Hence, the norm constraints (III.1) are now replaced by

$$d_1 \geq 1 - c_\nabla, \qquad d_l \geq 3/2 - c_\nabla \quad \text{for } l \in [2, L], \qquad d_{L+1} \geq 0.$$

The scale of the perturbation numerator now scales as $\tilde{\delta} W_t^l = n^{-d} \begin{cases} \Theta(n^{-d_1} n \theta_\nabla) & l = 1, \\ \Theta(n^{-d_l} n \theta_\nabla) & l \in [2, L], \\ \Theta(n^{-d_{L+1}} n^{-1}) & l = L+1. \end{cases}$

In μP, achieving maximal stable perturbations (V.4) is therefore equivalent to satisfying the constraints

$$d + d_1 = 0, \qquad d + d_l = 1 \quad \text{for } l \in [2, L], \qquad d + d_{L+1} = 0.$$

Now we can simultaneously satisfy the first- and last-layer norm constraints with $d_1 = 0$ and $d_{L+1} = 0$, while achieving effective perturbations in all layers with $d = 0$ and $d_l = 1$. Satisfying the norm constraint in the hidden layers with $d_l = 1/2$ would imply vanishing perturbations in the first and last layer (by requiring $d \geq 1/2$).

V.6. Representing general architectures and adaptive optimizers as a Tensor Program

Here, we lay out explicitly how to write some of the building blocks in ResNets and ViTs in a Tensor Program and provide further scaling considerations. According to Yang and Hu [52], it is straightforward to generalize scaling conditions that induce feature learning in MLPs to these other common neural network building blocks. Since perturbations should always scale like updates, the conditions for stable feature learning and those for stable effective perturbations are analogous.

Layernorm. The Layernorm operation is defined as

$$h_t^{l+1} = \gamma_t^l \frac{x_t^l - \nu_t^l}{\sigma_t^l + \varepsilon} + \beta_t^l,$$

where $\varepsilon > 0$ is a small positive constant, γ_t^l, β_t^l are learnable parameters and $\nu_t^l = \frac{1}{n}\sum_{i=1}^n (x_t^l)_i$ is an Avg operation as in Yang and Littwin [53, Def. 2.6.1] and $\sigma_t^l = \sqrt{\frac{1}{n}\sum_{i=1}^n (x_t^l - \nu_t^l)^2}$ is a composition of Nonlin, Avg and Nonlin. The parameters γ_t^l, β_t^l can be seen as input weights to the input 1. They should be initialized as $\gamma_0^l = 1$ and $\beta_0^l = 0$. In the forward pass, the layernorm preserves stability $h_t^{l+1} = \Theta(1)$ when $\gamma_t^l + \beta_t^l = \Theta(1)$ except for the Lebesgue nullset of learning rates for which they exactly cancel each other out. The derivatives are

$$d\beta_t^l = dh_t^{l+1}, \qquad d\gamma_t^l = dh_t^{l+1}\frac{x_t^l - \nu_t^l}{\sigma_t^l + \varepsilon}.$$

Using $\frac{\partial \sigma_t^l}{\partial x_t^l} = \frac{x_t^l - \nu_t^l}{n\sigma_t^l}$, we get

$$\begin{aligned} dx_t^l = & \quad dh_t^{l+1}\gamma_t^l\left(\frac{1}{\sigma_t^l + \varepsilon}(I - \frac{1}{n}) - \frac{x_t^l - \nu_t^l}{(\sigma_t^l + \varepsilon)^2}\frac{\partial \sigma_t^l}{\partial x_t^l}\right) \\ = & \quad dh_t^{l+1}\gamma_t^l\left(\frac{1}{\sigma_t^l + \varepsilon}(I - \frac{1}{n}11^T) - \frac{x_t^l - \nu_t^l}{(\sigma_t^l + \varepsilon)^2}\frac{(x_t^l - \nu_t^l)^T}{n\sigma_t^l}\right), \end{aligned}$$

which preserves the order as long as $\gamma_t^l = \Theta(1)$, since $x_t^l = \Theta(1)$, we know $\nu_t^l, \sigma_t^l = \Theta(1)$.

Note that Layernorm removes the necessity to avoid blowup in the activations x_t^l in the forward pass (ignoring potential numerical issues), and always rescales to $\Theta(\max(\gamma_t^l, \beta_t^l))$. However, in the backward pass, a scaling $x_t^l = \Theta(n^c)$, with $c > 0$, results in $dx_t^l = \Theta(n^{-c} dh_t^{l+1}\gamma_t^l)$, hence vanishing gradients. The gradients would only stabilize if $\phi'(h_t^l) = \Theta(h_t^l)$, but no popular activation function has a scale equivariant derivative.

Convolutions. Convolutional layers can be seen as a collection of dense weight matrices where width corresponds to the number of channels [50]. With kernel positions ker, input channels $[n^l]$ and output channels $[n^{l+1}]$, the weights of a stride-1 convolution are given by $\{W_{i\alpha\beta}^l\}_{i\in ker, \alpha\in[n^{l+1}], \beta\in[n^l]}$, so that for each $i \in ker$, $W_i^l \in \mathbb{R}^{n^{l+1}\times n^l}$ is a dense matrix. With $\{x_{i\alpha}^l\}_{i\in pos^l, \alpha\in[n^l]}$, the convolution operation is given by

$$(W^l * x)_{i\alpha} = \sum_{\beta, j: j+i\in pos^l} W_{j\alpha\beta}^l x_{i+j,\beta}^l,$$

which performs MatMul and Avg and where ker, pos^l are assumed to be of fixed size.

Residual connections. A residual connection propagates the current activation forward, skipping an arbitrarily complex nonlinear block $f_t^l : \mathbb{R}^{n_l} \to \mathbb{R}^{n_{l+1}}$ in between, where f_t^l can depend on time-dependent parameters like a weight matrix. The forward pass can be written as

$$x_t^l = x_t^{l-1} + f_t^l(x_t^{l-1}).$$

If $x_t^l = \Theta(1)$ for all layers l holds in the model without residual connections, it also holds in the model with residual connections. Note that as long as $x_t^1 = \Theta(1)$ and $f_t^l = O(1)$, it holds that $x_t^l = \Theta(1)$ for all l. But $f_t^l = o(1)$ should still be avoided, as it would hold that $x_t^{l+1} = x_t^l$ in the infinite-width limit and the layer would be superfluous. The derivative of the activations becomes

$$dx_t^{l-1} = dx_t^l + dx_t^l \frac{\partial f_t^l}{\partial x_t^{l-1}},$$

where the second term stays the same as without the residual connection. For the example of f_t^l being a fully connected layer we get $dx_t^{l-1} = dx_t^l + (W_t^l)^T \left(dx_t^l \odot \phi'(W_t^l x_t^{l-1}) \right)$. In this example, the derivative with respect to the weights becomes

$$\frac{\partial f_t}{\partial W_t^l} = dx_t^l \frac{\partial x_t^l}{\partial W_t^l} = dx_t^l \frac{\partial f_t^l}{\partial W_t^l} = (dx_t^l \odot \phi'(W_t^l x_t^{l-1}))(x_t^{l-1})^T,$$

where the residual connection does not alter the functional dependence on dx_t^l and x_t^l compared to a MLP, but implicitly influences the weight gradient since dx_t^l and x_t^l are altered. As for the forward pass, the gradient scaling dx_t^l gets stabilized in the backward pass so that $\frac{\partial f_t^l}{\partial x_t^{l-1}}$ is now allowed to be vanishing with width. Again, we are not aware of an architecture in which that would be desirable.

ADAM as a base optimizer. When using ADAM or similar adaptive optimizers as a base optimizer, the learning rate should scale as $\Theta(1)$ for input-like layers and biases, and $\Theta(n^{-1})$ for hidden and output layers [54]. Yang et al. [56] provide proofs for arbitrary optimizers that perform generalized, nonlinear outer products. In the example of ADAM, the update rule can be written as

$$\phi(u_\alpha^1, \dots, u_\alpha^k, v_\beta^1, \dots, v_\beta^k) = \sum_i \gamma_i u_\alpha^i v_\beta^k / \left(\sum_i \omega_i (u_\alpha^i v_\beta^i)^2 \right)^{1/2},$$

where γ_i, ω_i are the weights that stem from the moving averages. By using a learning rate of n^{-1} and using the fact that both u and v have approximately iid coordinates of order $\Theta(1)$, the law of large numbers yields $\Theta(1)$ updates of the form

$$\frac{1}{n} \sum_{\beta=1}^{n} \phi(u_\alpha^1, \dots, u_\alpha^k, v_\beta^1, \dots, v_\beta^k) x_\beta = \mathbb{E}\phi(u_\alpha^1, \dots, u_\alpha^k, Z^{v^1}, \dots, Z^{v^k}) Z^x.$$

Any other learning rate scaling would either result in blowup or vanishing updates.

V.7. Influence of width-dependent weight multipliers on bcd-parameterizations

Our definition of bcd-parameterizations is convenient because it purely adapts the learning algorithm but not the architecture. **We can also adapt the architecture by using layerwise width-dependent weight multipliers to avoid the necessity of layerwise perturbation scaling for effective perturbations in all layers.** The reason is that weight multipliers scale the gradients. In this section, we study how the introduction of weight multipliers affects bcd-parameterizations.

In this section, we consider L-hidden layer MLPs with weight multipliers $\{a_l\}_{l \in [L+1]}$, width $n \in \mathbb{N}$, inputs $\xi \in \mathbb{R}^{d_{\text{in}}}$, and with outputs $f(\xi) := n^{-a_{L+1}} W^{L+1} x^L(\xi)$ where the activations $x^L(\xi)$ are defined via the iteration

$$h^1(\xi) := n^{-a_1} W^1 \xi, \qquad x^l(\xi) := \phi(h^l(\xi)), \qquad h^{l+1}(\xi) := n^{-a_l} W^{l+1} x^l(\xi).$$

We define $abcd$-parameterizations in the same way as bcd-parameterizations, but instead of MLPs we use MLPs with weight multipliers $\{a_l\}_{l \in [L+1]}$.

$abcd$-equivalence classes. The weight multipliers n^{-a_l} affect the gradient scaling in the respective layer analytically as n^{-2a_l}. This can be counteracted by adapting the learning rate scaling. For

abc-parameterizations and SGD training, this induces the layerwise equivalence between parameterizations with (a_l, b_l, c_l) or with $(a_l + \theta, b_l - \theta, c_l - 2\theta)$. As perturbations are gradient-based, they behave like updates, and $d_l - 2\theta$ analytically cancels out the changed gradient scaling.

The following lemma formalizes the parameterization equivalences for SAM with SGD as a base optimizer. Its proof is provided at the end of this section. The extension to ADAM as a base optimizer is straightforward, since learning rate scalings and perturbation scalings are decoupled. For ADAM, c_l should be adapted to $c_l - \theta_l$.

Lemma 37 ($abcd$-equivalence classes) *Let $f_t(\xi)$ denote the output of a MLP in a stable $abcd$-parameterization with weight multipliers $\{a_l\}_{l\in[L+1]}$ after t steps of training with the SAM update rule with layerwise perturbation scaling (LP) using a fixed sequence of batches and evaluated on input ξ. Then for any $\{\theta_l\}_{l\in[L+1]} \subset \mathbb{R}$, $f_t(\xi)$ stays fixed for all t and ξ if*

(a) $\{d_l\}_{l\in[L+1]}$ is reparameterized to $\{d_l + C\}_{l\in[L+1]}$ for some arbitrary $C \in \mathbb{R}$,
(b) (a_l, b_l, c_l, d_l, d) is reparameterized to $(a_l + \theta_l, b_l - \theta_l, c_l - 2\theta_l, d_l - 2\theta_l, d)$.

Impacts on global perturbation scaling. Here we restrict ourselves to the μP equivalence class of abc-parameterizations, and do not allow layerwise perturbation scaling. We are interested in the maximal stable choice of global perturbation scaling n^{-d} to at least achieve non-vanishing perturbations in some layers. Note that there exist multiple $abcd$-equivalence classes that satisfy μP, but they induce differing global perturbation scalings. The crucial difference between SAM and SGD is that the layers are coupled through the joint gradient normalization in the update rule. Different choices of $\{a_l\}_{l\in[L+1]}$ induce different relative scalings of the gradient norms, so that different subsets of the layers dominate the total gradient norm, and allow different choices of global perturbation scaling d. Therefore the maximal global perturbation scaling will not always belong to the same equivalence class. This even affects which subset of layers will be effectively perturbed in global perturbation scaling. The following lemma summarizes the effect of the choice of weight multipliers $\{a_l\}_{l\in[L+1]}$ on the $abcd$-parameterization with maximal stable perturbations. Its proof is provided at the end of this section.

Lemma 38 (Effect of weight multipliers on global scalings) *Assume $\{(a_l, b_l, c_l)\}_{l\in[L+1]}$ are chosen from the μP equivalence class. Assume there is some $C \in \mathbb{R}$ such that $d_l = C$ for all $l \in [L+1]$.*

*(a) **(Naive perturbation scaling is not sufficient)** Under naive perturbation scaling $d = 0$, there is no choice of $\{a_l\}_{l\in[L+1]}$ that achieves effective perturbations in the first layer.*
*(b) **(Global scaling can effectively perturb all layers)** Global perturbation scaling achieves effective perturbations in all layers as long as the following constraints are satisfied,*

$$d = -1/2, \qquad a_l = a_1 + 1/2 \text{ for } l \in [2, L], \qquad a_{L+1} = a_1 + 1.$$

*(c) **(`mup` package global scaling effectively perturbs hidden layers)** The `mup`-package implements the choice $a_{L+1} = 1$ and $a_l = 0$ for $l \in [L]$. Then maximal stable perturbations are achieved under $d = 0$. In this parameterization, hidden layers are effectively perturbed, but first and last layers are not effectively perturbed. μP^2 would now be achieved with the choice $d = d_1 = d_{L+1} = -1/2$ and $d_1 = -1/2$, $d_l = 1/2$ for $l \in [2, L]$.*

The above lemma asks how much we can simplify the perturbation scaling while recovering effective perturbations in all layers. Lemma 38(a) shows that the global perturbation radius always has to be scaled nontrivially, in order to achieve effective perturbations in all layers. The reason is that the gradient normalization prevents the weight multipliers from arbitrarily scaling the global perturbation radius. But Lemma 38(b) shows that with global perturbation scaling $d = -1/2$ layerwise perturbation scaling is not necessary, with a proper choice of weight multipliers. One choice of $\{a_l\}_{l\in[L+1]}$ that achieves such *effective global perturbation scaling* would be $a_1 = 0$, $a_l = 1/2$ for $l \in [2, L]$ and $a_{L+1} = 1$. This choice resembles the `mup`-package if one implemented hidden layers with weight multiplier $n^{-1/2}$. The current implementation of the `mup`-package seems suboptimal for the SAM update rule with layerwise perturbation scaling (LP) in the sense that it requires both an adaptation of the architecture and of the learning algorithm, whereas bcd-parameterizations only adapt the learning algorithm and $abcd$-parameterizations from Lemma 38(b) only require adapting the architecture and the global perturbation radius. The generalized gradient norm and effective perturbation constraints in the proof of Lemma 38 jointly show that in all $abcd$-parameterizations that achieve effective perturbations in all layers, $d = -1/2$ and only the first layer contributes non-vanishingly to the gradient norm.

Proof of Lemma 37

Part (a) is immediately implied by the normalization of v_t^l in (LP).

Part (b) follows in the same way as for SGD Yang et al. [54]. Analytically, $a_l + \theta$ scales the l-th layer's gradient as $n^{-2\theta}$, which can be counteracted by $c_l - 2\theta$ and $d_l - 2\theta$, since both the c_l and d_l scale analogous gradients.

■

Proof of Lemma 38

In general, in the abc-equivalence class of μP, the l-th layer's gradient norm is scaled by n^{-2a_l}. This induces the generalized gradient norm constraints for $\|\nabla_W L\| = \Theta(1)$,

$$d_1 \geq -1/2 - 2a_1, \qquad d_l \geq -2a_l, \qquad d_{L+1} \geq 1/2 - 2a_{L+1}.$$

Effective perturbations are achieved when $\rho n^{-d-d_l}\nabla_{W^l} L = \Theta(n^{-\mathbb{I}(l>1)})$, which induces the perturbation stability constraints

$$d + d_1 \geq -1 - 2a_1, \qquad d + d_l \geq -2a_l, \qquad d + d_{L+1} \geq 1 - 2a_{L+1},$$

with effective perturbations whenever the equality of the respective layer holds.

Part (a):

Under naive perturbation scaling $d = 0$, it is clear that the first layer gradient norm constraint $d_1 \geq -1/2 - 2a_1$ is exhausted before effective perturbations are achieved at $d_1 = -1 - 2a_1$.

Part (b):

For some $C \in \mathbb{R}$, the gradient norm constraints become $C \geq -1/2 - 2a_1, C \geq -2a_l, C \geq 1/2 - 2a_{L+1}$, where one equality is required to hold. The effective perturbation constraints become $d + C \geq -1 - 2a_1, d + C \geq -2a_l, d + C \geq 1 - 2a_{L+1}$. Hence, if the last layer gradient norm constraint is fulfilled, $d = 1/2$ achieves effective last-layer perturbations, but first and hidden layers

cannot be effectively perturbed; $d < 1/2$ then induce output perturbation blowup. But, if the first-layer norm constraint is fulfilled (by $C = -1/2 - 2a_1$), then $d = -1/2$ induces effective first-layer perturbations, and the choices $a_l = 1/2 + a_1$ and $d_{L+1} = 1 + a_1$ achieve effective perturbations in hidden and last layers while contributing vanishingly to the gradient norm.

Part (c):

The `mup`-package implements μP for the last layer with the width multiplier n^{-1}, the initialization variance $N(0, 1)$ and the SGD learning rate ηn, letting them treat all weight vectors (input-like or output-like) in the same way. As updates scale as $\eta n \nabla_{W^{L+1}} L = \Theta(n^{-1})$, the last-layer gradient must now scale as $\nabla_{W^{L+1}} L = \Theta(n^{-2})$ and $\|\nabla_{W^{L+1}} L\| = \Theta(n^{-3/2})$ (this could also be derived by elementary calculations). Hence the last-layer gradient norm constraint (III.1) is replaced by $d_{L+1} \geq -\frac{3}{2}$ with equality for $\Theta(1)$ gradient norm contribution. For effective perturbations in the last layer, we need $\rho n^{-d-d_{L+1}} \nabla_{W^{L+1}} L = \Theta(n^{-1})$, which translates into $d + d_{L+1} = -1$ for effective perturbations, where $d + d_{L+1} \geq -1$ is required for perturbation stability. The overall weight scaling $W^{L+1} = \Theta(n^{-1})$ remains invariant, and with it $dx^L = W^{L+1}$ and the gradients of all previous layers. To summarize, the adapted norm constraints for $n^{-d_l} \|\nabla_{W^l} L\| = \Theta(1)$ in `mup` are now

$$d_1 \geq -1/2, \qquad d_l \geq 0, \qquad d_{L+1} \geq -3/2,$$

and the conditions to achieve effective perturbations $\tilde{\delta} W^l = \Theta(n^{-\mathbb{I}(l>1)})$ in `mup` are

$$d + d_1 \geq -1, \qquad d + d_l \geq 0, \qquad d + d_{L+1} \geq -1.$$

Maximal perturbations in all layers are now achieved with the previous choices $d = -1/2$, $d_1 = -1/2$, $d_l = 1/2$, but now $d_{L+1} = -1/2$ in the last layer, which still contributes vanishingly to the gradient norm.

For global scaling, now all hidden layers contribute to $\Theta(1)$ gradient norm with $d_l = 0$ for all $l \in [L + 1]$. The minimal stable choice of global perturbation scaling is now $d = 0$ achieving effective hidden-layer perturbations, but not effective first- or last-layer perturbations.

■

V.8. Implementation of the spectral μP perspective for varying widths

When allowing the width to vary inside the network, we need to track the initialization, learning rate and perturbation scaling more carefully in terms of layerwise width n_l. As varying width is common, this is the version of bcd-parametrizations we implement. We closely follow the implementation in the `mup` package by introducing a base width at which SP and μP are equivalent, allowing to immediately transfer setups that perform well in SP.

While we use the `mup` package as a basis for the ViT experiments, for MLPs and ResNets we do not introduce a width multiplier in the output layer and directly implement *bcd*-parameterizations using the spectral conditions from Yang et al. [55]. Yang et al. [55] provide a simple key requirement on the spectral norm scaling of weight matrices to achieve μP. With input width `fan_in` and output width `fan_out`, the initialization variance should be chosen as $b = \frac{1}{\sqrt{\texttt{fan\_in}}} \min\left\{1, \sqrt{\frac{\texttt{fan\_out}}{\texttt{fan\_in}}}\right\}$, and learning rate $c = \frac{\texttt{fan\_out}}{\texttt{fan\_in}}$.

Implementation details for μP^2. We scale layerwise learning rates with the layerwise factor $\frac{\texttt{fan\_out}}{\texttt{fan\_in}}$. For ADAM, due to gradient normalization, the learning rate should be scaled as $\frac{1}{\texttt{fan\_in}}$. The initialization variance remains the same $\frac{1}{\texttt{fan\_in}}$ in all except the last layer, which we initialize to 0 anyways. The layerwise perturbation radii should scale as the learning rates $\frac{\texttt{fan\_out}}{\texttt{fan\_in}}$. If we decouple the gradient normalization terms in the denominator of the SAM update rule and intend to scale them all to $\Theta(1)$, each input-like and each hidden-like term should be scaled as $\sqrt{\frac{\texttt{fan\_out}}{\texttt{fan\_in}}}$. Here output layers require a special treatment, if we care about the correct width-independent spectral constants, which may be significant when n_{L+1} is large, as in ImageNet1K. For hidden and input layers $l \in [L]$, we get $dW^l = dh^l(x^{l-1})^T = \Theta(1/n_l)$, where $dh^l = \Theta(1/n_l)$ coordinatewise with Frobenius (equivalently spectral) norm $\|dW^l\|_* = \|dW^l\|_F = \sqrt{\frac{n_{l-1}}{n_l}}$. Output layer gradients scale as $dW^{L+1} = x^L = \Theta(1)$ thus $\|dW^{L+1}\|_* = \|dW^{L+1}\|_F = \sqrt{n_L n_{L+1}}$. This also shows that fulfilling the spectral update condition $\|\Delta W^{L+1}\|_* = \Theta(\sqrt{\frac{n_{L+1}}{n_L}})$ strictly speaking requires a SGD learning rate scaling $\frac{1}{n_L}$. For biases before the last layer, $db^l = dh^l$ so $\|db^l\|_* = \|db^l\|_F = \sqrt{\frac{1}{n_l}}$, which fits into the scheme $\sqrt{\frac{\texttt{fan\_in}}{\texttt{fan\_out}}}$, but for the output layer bias $\|db^{L+1}\|_F = \|1\|_F = \sqrt{n_{L+1}}$. In the output layer, the term $dh^l = \Theta(1/n_l)$ is missing. The output multiplier used in the `mup` package does not correct this difference, but using a base width cancels out all occurences of n_{L+1}.

For elementwise ASAM, it suffices to rescale the global perturbation radius by $\sqrt{n_L}$, assuming all width dimensions scale at the same rate.

For layerwise ASAM and $l \in [L]$, we get $\|W_t^l\|_F = \|W_0^l\|_F = \Theta(\sqrt{n_l})$, which requires layerwise perturbation scaling $\frac{1}{\texttt{fan\_in}}$, and in the denominator $\|W^l\|_F \|\nabla_{W^l} L\|_* = \sqrt{n_l}\sqrt{\frac{n_{l-1}}{n_l}} = \sqrt{n_{l-1}}$ requires $\sqrt{\frac{1}{\texttt{fan\_in}}}$ for width independence. For output layer weights $\|W^{L+1}\|_F^2 = \|\Delta W^{L+1}\|_F^2 = \Theta(\frac{n_{L+1}^3}{n_L})$. For perturbations that fulfill the spectral condition $\rho_{L+1}\|W^{L+1}\|_F^2 \|\nabla_{W^{l+1}} L\|_* = \Theta(\sqrt{\frac{n_{L+1}}{n_L}})$, we need to choose $\rho_{L+1} = \frac{1}{n_{L+1}^3}$ (width-independent). The last-layer denominator term scales as $\|W^{L+1}\|_F \|\nabla_{W^{l+1}} L\|_* = \Theta(\sqrt{\frac{n_{L+1}^3}{n_L}} \cdot \sqrt{n_L n_{L+1}}) = \Theta(n_{L+1}^2)$, which is width independent, but can be a big constant if n_{L+1} is large like in ImageNet1K. The output bias has $\|\Delta b^{L+1}\|_F^2 = n_{L+1}$ requiring $\rho = \Theta(\frac{1}{\texttt{fan\_out}})$. Using a base width, the scalings of both output weights and the output bias become width-independent.

If using the `mup`-package, only the last layer scalings change. W^{L+1} and ΔW^{L+1} remain invariant, but $\nabla_{W^{L+1}} L$ should be scaled by n_L^{-2} everywhere. For SAM, the last layer can now be scaled like input layers. More details can be found in Appendix V.7. For SAM-ON nothing changes, as only input-like layers are perturbed. For elementwise ASAM, the last-layer numerator scaling becomes $\Theta(n_L^{5/2})$, and the last-layer denominator scaling becomes $\Theta(n_L^{-2})$, assuming a base width and ignoring the fixed n_{L+1}. For the example of layerwise ASAM, we get $\|W^{L+1}\|_F \|\nabla_{W^{l+1}} L\|_* = \Theta(\frac{n_{L+1}^2}{n_L^2})$, and $\rho_{L+1} = \frac{n_L^2}{n_{L+1}^3}$.

To provide a code example, the SAM implementation Samuel [41] can simply be adapted to μP^2 as follows (scaling the gradient norm contributions of all layers to $\Theta(1)$ as we do for the ViT experiments). Note that while SP cannot induce effective perturbations (as proven by Theorem 4),

one can still run SP with layerwise perturbations (SP-MPP) empirically. The crucial parameters to track are `group["rho"]` and `group["gradnorm_scaling"]`:

```
import math, torch
from mup import MuAdamW

# specify parameterization
parameterization, perturbation = 'mup', 'mpp'

# specify model and hyperparameters
model, lr, rho, weight_decay, last_layer_weight_name = ...


class SAM(torch.optim.Optimizer):
  ...

  def grad_norm(self):
    grads = []
    for i, group in enumerate(self.param_groups):
      for p in group["params"]:
        grads.append((group["gradnorm_scaling"] * p.grad).norm(p=2))
    norm = torch.stack(grads).norm(p=2)
    return norm

  @torch.no_grad()
  def first_step(self): # the perturbation step before the weight update
    grad_norm = self.grad_norm()
    for group in self.param_groups:
      scale = group["rho"] / (grad_norm + 1e-12)
      for p in group["params"]:
        if p.grad is None: continue
        self.state[p]["old_p"] = p.data.clone()
        e_w = p.grad * scale.to(p)

        p.add_(e_w)  # climb to the local maximum "w + e(w)"


# definition of the SAM optimizer for:
# LR parameterization (parameterization in ['sp', 'mup']), and
# RHO parameterization (perturbation in ['naive','global','mpp'])

param_groups = []
if parameterization == 'mup' or perturbation == 'mpp':
  for name, p in model.named_parameters():
    if p.infshape.ninf() == 0 or perturbation in ['global','naive']:
      group = {
        "params": [p],
        "lr": lr,
        "rho": rho,
        "gradnorm_scaling": 1,
      }
    elif p.infshape.ninf() == 1:
      # vector-like
      for d in p.infshape:
        if d.base_dim is not None:
          factor = d.dim / d.base_dim
```

```
          break
      group = {
        "params": [p],
        "lr": lr,
        "rho": rho * factor,
        "gradnorm_scaling": math.sqrt(factor),
      }
      if name == last_layer_weight_name:
        group["gradnorm_scaling"] = math.sqrt(factor)**3
    elif p.infshape.ninf() == 2:
      # matrix-like
      factor = (p.infshape[0].dim/p.infshape[1].dim) * (p.infshape[1].base_dim/p
    .infshape[0].base_dim)
      group = {
        "params": [p],
        "lr": lr,
        "rho": rho * factor,
        "gradnorm_scaling": math.sqrt(factor),
      }
    else:
      raise NotImplementedError
    param_groups.append(group)
elif parameterization == 'sp':
  for name, p in model.named_parameters():
    group = {
      "params": [p],
      "lr": lr,
      "rho": rho,
      "gradnorm_scaling": 1,
    }
    param_groups.append(group)
else:
  raise NotImplementedError


optimizer = SAM(param_groups,
  base_optimizer=MuAdamW if parameterization=='mup' else torch.optim.AdamW,
  weight_decay=weight_decay)
```

Algorithm 1: Implementation of μP^2 for SAM using the `mup`-package and PyTorch. This code scales the gradient norm contributions of all layers to be width-independent.

VI. Experimental details

If not mentioned otherwise, experiments use the settings specified in this section. While we directly implement *bcd*-parameterizations for MLPs and ResNets in PyTorch [39], we use the `mup`-package [54] as a basis for ViT experiments.

MLPs. We train 3-layer MLPs without biases with ReLU activation function for 20 epochs with constant learning rate, using SGD as base optimizer as specified in Definition 3, but allow for SGD batchsize larger than 1, defaulting to batch size 64. We evaluate the test accuracy after every epoch and use the snapshot across training with the best accuracy. This is necessary as the test accuracy

Hyperparam.	SAM		SAM-ON		ResNet18	Elem. ASAM		Layer ASAM	
	SP	μP^2	SP	μP^2	SGD	SP	μP^2	SP	μP^2
Training epochs					200				
Batch size					64				
LR η	0.05	2^{-4}	0.05	2^{-4}		0.05	2^{-4}	0.1	2^{-4}
LR decay					Cosine				
Weight decay					0.0005				
Momentum					0.9				
Labelsmoothing					0.1				
Pert. radius ρ	0.1	2^{-4}	0.5	$5 \cdot 2^{-4}$		2	$10 \cdot 2^{-4}$	0.02	2^{-6}
Output multiplier	1	0.125	1	0.125		1	0.125	1	0.125

Table VI.1: **(ResNet-18 hyperparameters for CIFAR10)** Hyperparameters for SP are taken from Müller et al. [35]. Learning rate and perturbation radius are tuned using the experiments in Appendix VII.3.2. ResNets in μP have base width 0.5, gradient norm scaling according to Definition 3 and their last layer is initialized to 0.

is not monotonically increasing across training, while the training accuracy is. For ResNets we do not observe such harmful overfitting. For the standard parametrization, we use He initialization [19] and don't tune multipliers to mimic standard training procedures. For μP, we resort to the optimal multipliers from Yang et al. [54]. We then find the optimal learning rate and perturbation radius for each bcd-parametrization and SAM variant separately.

ResNets. For ResNet18 experiments, we augment the CIFAR10 data with random crops and random horizontal flips, set labelsmoothing to 0.1 and use a cosine learning rate schedule. ResNets in μP have base width 0.5, gradient norm scaling according to Definition 3 and their last layer is initialized to 0. For SP, we again adopt the standard hyperparameters from Müller et al. [35] by using a momentum of 0.9, weight decay 0.0005, an output multiplier of 1.0, and individually tuned learning rate and perturbation radius for each SAM variant. For μP, at base width multiplier 0.5 compared to the original width, for each SAM variant, we perform a random grid search over the hyperparameters learning rate, perturbation radius, output multiplier $[2^{-8}, 2^{-7}, \dots, 2^{8}]$, weight decay $[0, 10^{-5}, 10^{-4}, 5 \cdot 10^{-4}, 10^{-3}, 10^{-2}]$ and momentum $[0, 0.1, 0.4, 0.7, 0.9]$. Learning rate and perturbation radius grids were either set to $[2^{-10}, 2^{-9}, \dots, 2^{1}]$ or centered around recommendations from the literature. The optimal hyperparameter configurations found from at least 150 runs for each SAM variant are summarized in Table VI.1. Learning rates and perturbation radii were further tuned with the experiments from Appendix VII.3.2.

ViTs. We train ViT-S/16 with 6 layers and 12 attention heads on ImageNet1K [12] and a ViT-S/4 with 12 layers and 12 attention heads on CIFAR100 [27] (see **??**), again adopting the hyperparameter settings from Müller et al. [35]. This means we use AdamW as a base optimizer with warmup and a cosine learning rate decay. For CIFAR100, we use random crops, random horizontal flips and AutoAugment as data augmentations. For Imagenet we use the original preprocessing from Huggingface `vit-base-patch16-224` [48]. For μP, we tune multipliers at a basewidth 384, initialize the last layer and query weights to 0. By using the μP package, the relative perturbation scalings change as explained in Appendix V.8 and Appendix V.7. Global and naive perturbation

Hyperparam.	SAM on ImageNet1K			SAM on CIFAR100	
	SP	μP^2	shared	SP	μP^2
Training epochs	100			300	
Batch size	128				
LR η	0.001	0.00226		0.0005	
LR warmup epochs	10			30	
LR decay	Cosine				
Weight decay	0.1	0.0872		0.05	
Labelsmoothing	0.1				
Pert. radius ρ	1	1.1939		0.25	0.25
Input multiplier	1	1.7309		1	1.7309
Output multiplier	1	4.0946		1	4.0946
Layers	6			12	
Attention heads	12				
Patch size	16			4	

Table VI.2: **(Vision Transformer hyperparameters)** Hyperparameters for SP are taken from Müller et al. [35] using AdamW as a base optimizer. ViTs in μP have base width 384, last layer and query weights are initialized to 0 and gradient norm contributions of all layers are scaled to $\Theta(1)$.

	ResNet-18 on CIFAR10				ViT on CIFAR100			ViT on ImageNet1K		
Width multiplier	0.5	1	2	4	0.5	1	2	0.5	1	2
Seconds per epoch	109	161	327	803	209	327	777	2550	4151	9802

Table VI.3: **(Training time per epoch)** Training time (in seconds) per epoch of the entire data loading and training pipeline of SAM in μP^2 on a single NVIDIA A10G GPU.

scaling in μP now coincide. Here, instead of the original perturbation scaling Definition 3, we scale the gradient norm contributions of all layers in the denominator to $\Theta(1)$. The hyperparameter choices for ViTs on CIFAR100 and ImageNet are summarized in Table VI.2. For μP, the learning rate, perturbation radius, input multiplier, output multiplier and weight decay were tuned using 3 independent runs of Nevergrad `NGOpt` with budget 56 on ImageNet. The same multipliers are used on CIFAR100.

Figures. Whenever multiple runs with independent random seeds are used for training, confidence bands cover the interval from the empirical 2.5%- to the empirical 97.5%-quantile. The line then denotes the average of all runs. When confidence bands are given, but the number of independent runs is not specified, the number of runs defaults to 4.

Computational resources. We ran all of our experiments on Amazon EC2 G5 instances each containing up to 8 NVIDIA A10G GPUs. On a single GPU, our μP^2-SAM training script for MLPs of width 4096 on CIFAR10 takes 502 seconds to run in total (25 seconds per epoch), where data handling takes most of the time. The training times for ResNets and ViTs are presented in Table VI.3.

VII. Supplemental experiments

This section provides more extensive empirical evaluations to validate the claims of the main paper. By naive perturbation scaling (naive) we denote parameterizations that do not adapt any perturbation scalings ($d = d_l = 0$ for all l). Global perturbation scaling (global) denotes the maximal stable scaling n^{-d} of the global perturbation radius that achieves effective perturbations in some layers without layerwise perturbation scaling ($d_l = 0$ for all l).

While previous μP-literature mostly focuses on the more immediate transfer in training error, for SAM it is crucial to consider optimality in test error as the perturbation radius acts as a regularizer, so that optimality in test error typically coincides with suboptimal training error.

VII.1. SAM is approximately LL-SAM in μP with global perturbation scaling

Figures VII.1 and VII.2 compare SAM in μP under global perturbation scaling (μP-global) with SAM under global perturbation scaling where only the last-layer weights are perturbed (LL-SAM) by showing more neural network statistics that are related to SAM's inductive bias and to learning in general. From top-left to bottom right, the statistics are: Frobenius norm of the layerwise weight perturbation (which is closely related to spectral norm as perturbations are low rank); Frobenius norm of the layerwise weight perturbation normalized by the weight spectral norm to upper bound the influence of the perturbations on the output; spectral norm of the weight updates across training scaled by the spectral condition $n^{1/2}$, 1 and $n^{-1/2}$ for input, hidden and output layers respectively; norm of the activation updates for each layer normalized by the square root of the layer's output dimension to measure coordinatewise update scaling; layerwise effective feature ranks measured as in Andriushchenko et al. [2] by the minimal amount of singular values to make up 99% of the variance of the activations in a given layer; gradient norm, Hessian spectral norm and Hessian trace of loss with respect to weights; training accuracy, test accuracy after optimally stopping.

Observe that especially for large widths, global perturbation scaling effectively only perturbs the last layer, as predicted by Theorem 4. Last-layer SAM is more similar to μP-global SAM than SGD on all of the tracked statistics, in particular at large widths. Only perturbing the last layer still affects the gradients in earlier layers so that weight updates and activations change in all layers. We find that SAM in μP with global scaling does not consistently improve generalization performance over SGD, whereas μP^2 does improve over SGD for all widths (Figure VII.4). Last-layer perturbation norms coincide by design with the global perturbation radius $n^{-d}\rho$ and their effect on the activations stays $\Theta(1)$ with increasing width as measured in relation to weight spectral norm. Formally the last-layer perturbation norm converges due to

$$\|\tilde{W}^{L+1} - W^{L+1}\|_F = n^{-d}\rho \|\frac{\chi_t x_t^L}{\|v_t\|}\|_F \to n^{-d}\rho \|\frac{x_t^L}{\|x_t^L\|}\|_F = n^{-d}\rho \to 0,$$

where the loss derivative χ_t always cancels out due to the normalization and the global gradient norm $\|v_t\|$ is dominated by the last-layer gradient norm due to the global scaling (Theorem 4). Normalizing the weight perturbations by the weight spectral norm measures the influence of the perturbations on the activations. Note that this influence is also vanishing. Feature ranks stay close to initialization, since random initialization has high rank and training does low effective rank updates. Here we do not observe that SAM reduces the feature rank compared to SGD. The Hessian spectral norm

and trace are quite noisy. The last-layer Hessian spectral norm explodes with width in μP, because last-layer learning rate is scaled as n^{-1}, hence the edge of stability explodes. ResNets in μP are more stable, their Hessian spectral norm even shrinks with width (not shown).

Contrast the results for μP-global with the results for μP^2 in Appendix VII.2 for a comparison with SGD in μP. The Hessian spectral norm is reduced by SAM as you would expect. Additionally μP^2 shows low variability in performance and all other statistics. SAM in μP^2 does not reduce the feature rank compared to SGD in μP. This suggests that the conclusions drawn by Andriushchenko et al. [2] do not apply to MLPs in μP.

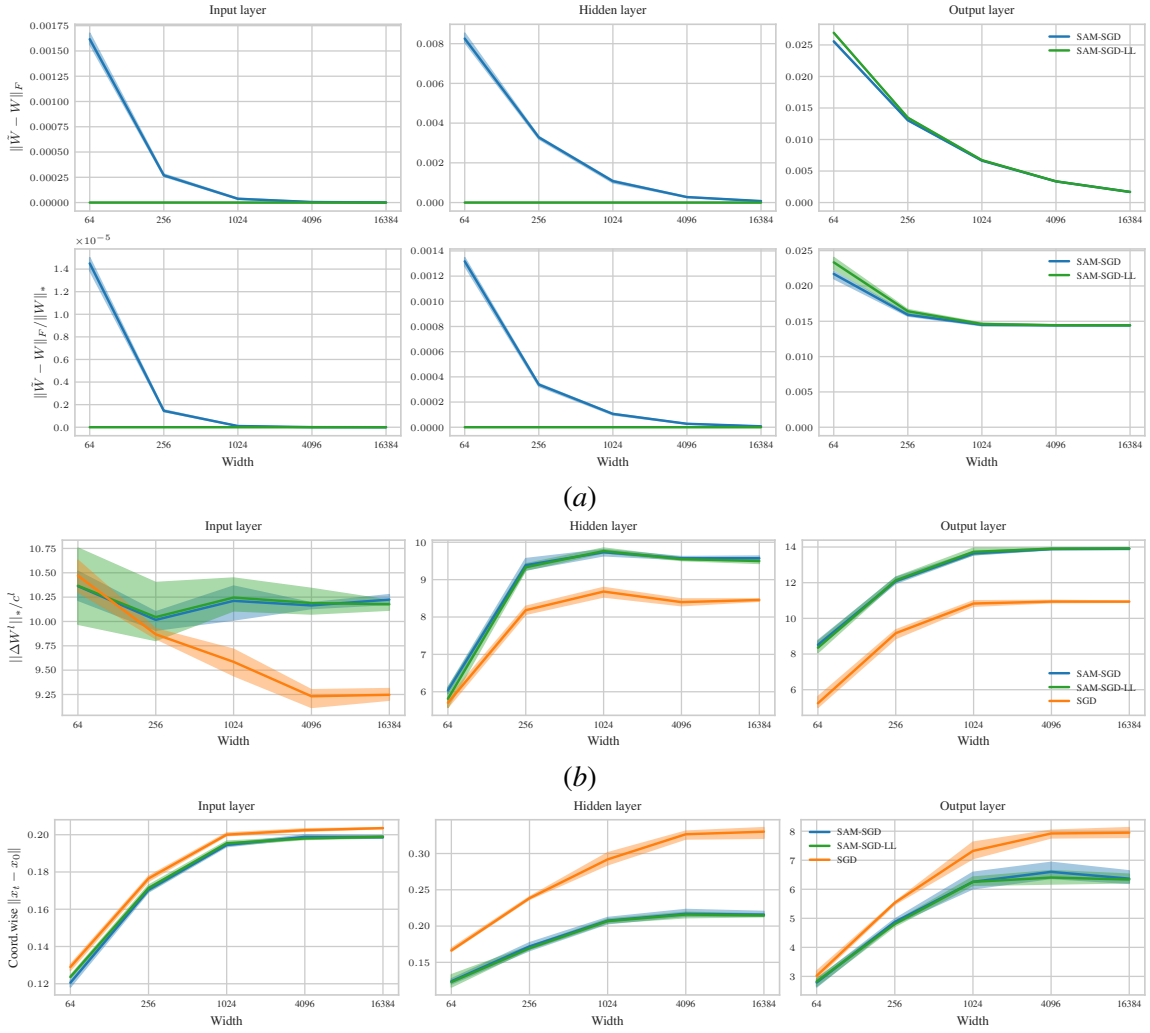


Figure VII.1: Several neural network statistics for SAM (blue), LL-SAM (green) and SGD as a baseline (orange) across width after training a 3-layer MLP in μP-global for 20 epochs with the optimal learning rate 0.3432 and perturbation radius 0.2154. The statistics are explained in the text of Appendix VII.1.

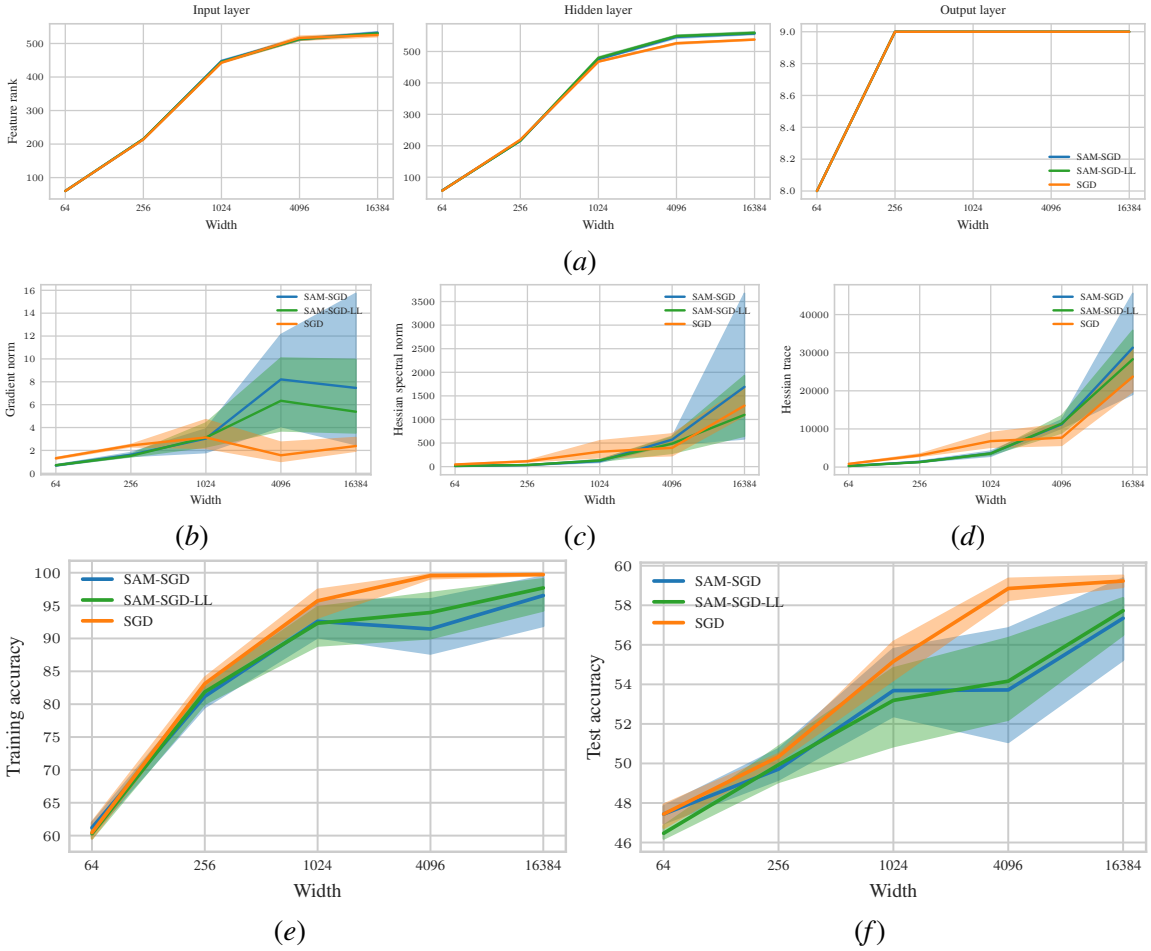
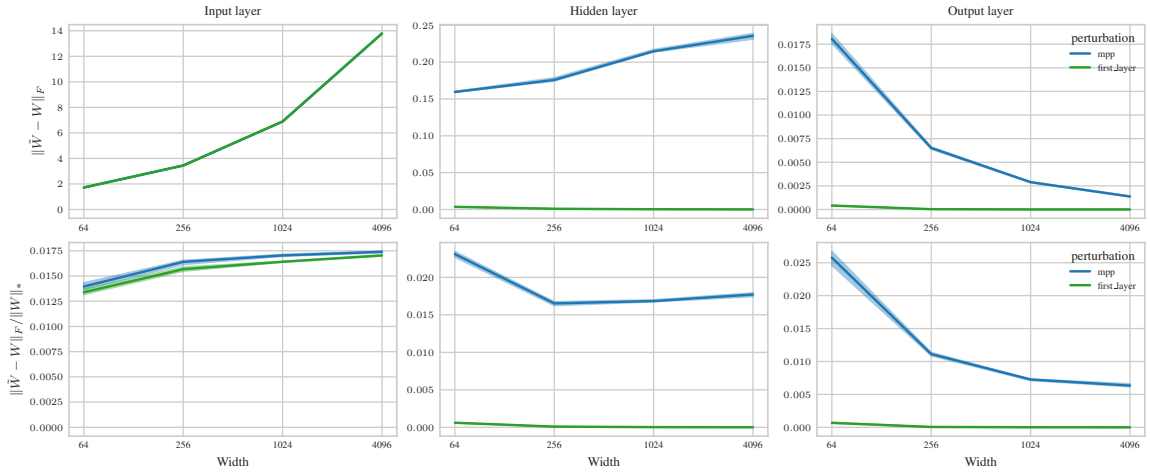


Figure VII.2: Figure VII.1 continued.

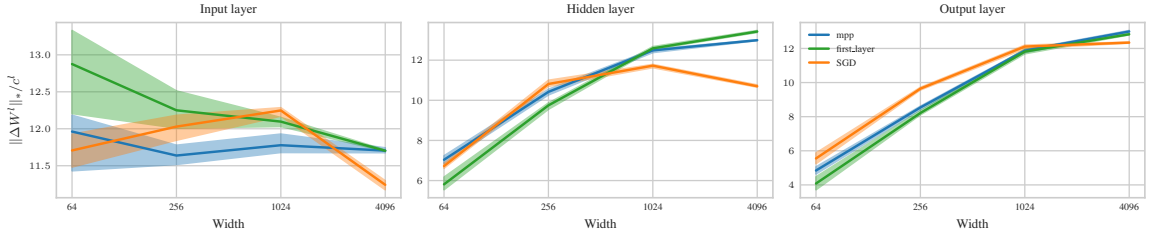
VII.2. Propagating perturbations from the first layer does not inherit SAM's benefits

Here we apply a parametrization that only effectively perturbs the first layer weights (derived in Example 1). Appendix VII.2 and Fig. VII.3 shows that effective first-layer SAM loses both μP^2 SAM's improvement in test accuracy as well as SAM's inductive bias towards smaller gradient norm and Hessian norm, i.e. lower sharpness in MLPs. This performance deterioration occurs although the perturbation of first-layer SAM has an effect of the same order of magnitude as μP^2 on weight and activation updates in all layers. This shows that mere propagation of weight perturbations from earlier layers cannot replace effective weight perturbations in each layer in order to benefit from SAM. It is crucial to correctly adjust the layerwise perturbation scaling, and to distinguish between effective perturbations and perturbation nontriviality in each layer.

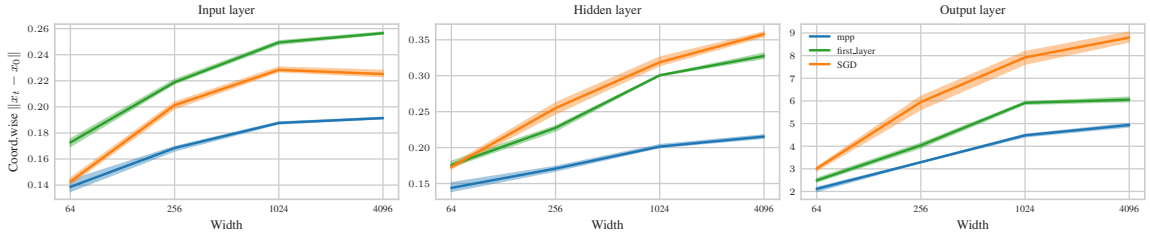
SAM in μP^2, on the other hand, achieves the correct perturbation and update scaling, has lower final gradient and Hessian spectral norm, improves test accuracy over SGD and has overall lower variance between training runs.



(*a*)



(*b*)



(*c*)

Same neural network statistics as in Figure VII.1 but SAM-SGD in μP^2 (blue) versus MUP with perturbations scaled to only effectively perturb the first layer weights (green) with SGD in μP as a baseline. The first-layer perturbation parameterization performs worse than μP^2 and results in gradient norm and Hessian norm similar to that of SGD, larger than those of SAM. While the spectral norm of the weights converges to a similar quantity as for μP^2, the effect of the weight changes on the hidden activation updates behaves more like SGD. Feature ranks all look similar.

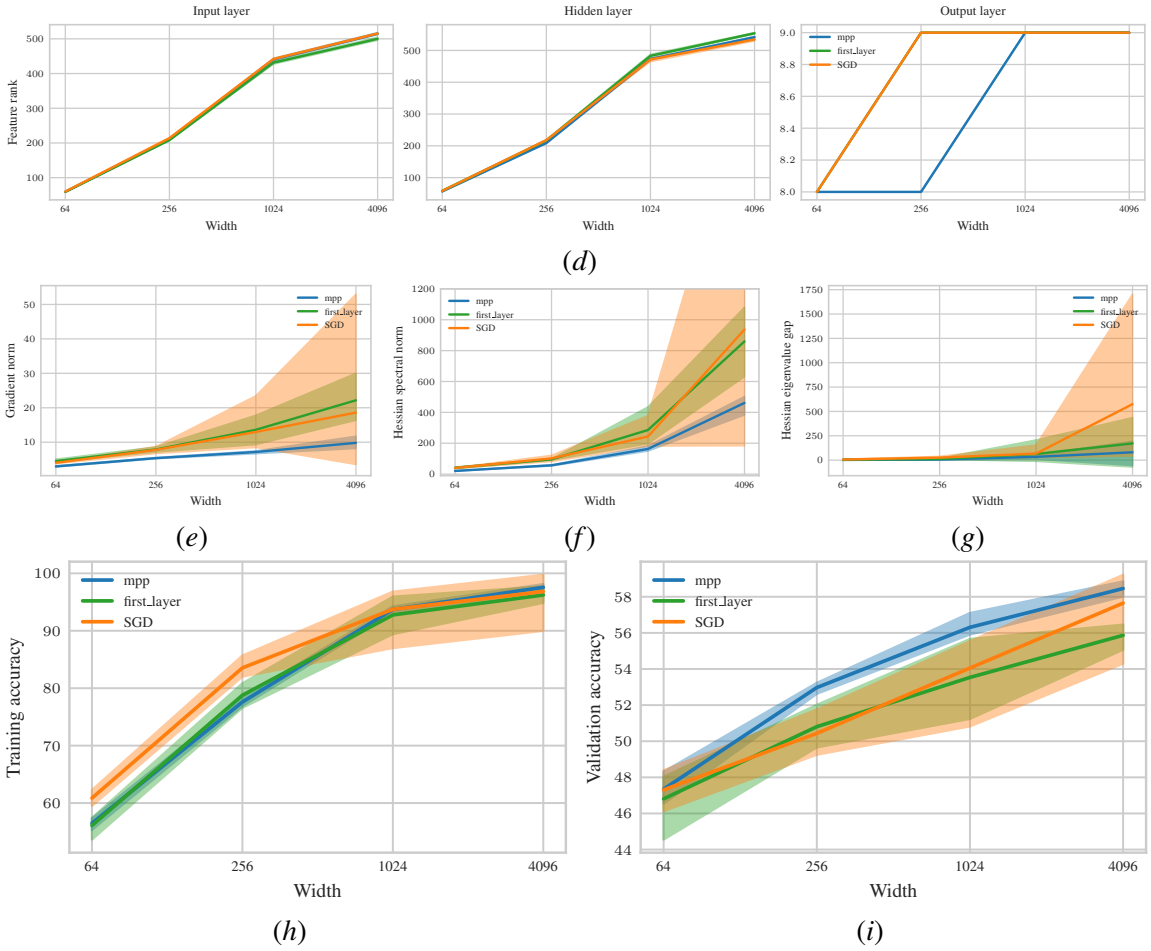


Figure VII.3: Appendix VII.2 continued.

VII.3. Hyperparameter transfer

In this section, we provide supplemental evidence that μP^2 is the unique parameterization that robustly achieves hyperparameter transfer both for the optimal learning rate and the optimal perturbation radius across neural architectures and datasets.

VII.3.1. MLPs

Figure VII.4 shows that in μP^2 the optimal hyperparameters in terms of test accuracy transfer in both learning rate and perturbation radius at sufficient width. In μP^2, SAM improves over SGD more than in SP, and the overall best test accuracy is achieved with the widest MLPs in μP^2.

While other works focus on hyperparameter transfer in training loss, we are ultimately interested in transfer with respect to test accuracy. Especially under harmful overfitting, the test accuracy is affected by nontrivial interactions between the learning rate and the perturbation radius. While

the joint optimum is slightly shifting towards larger learning rate and perturbation radius for small widths, it remains remarkably stable for sufficient width ≥ 1024. Note that slight shifts in the optimal learning rate due to finite width biases have also been observed in earlier works [54].

Figure VII.5 shows that global perturbation scaling does transfer the same perturbation instability threshold, whereas in μP-naive every fixed perturbation radius becomes unstable at sufficient width (Appendix VII.3.1). But in μP-global we do not observe a benefit of SAM over SGD. While the optimal learning rate with respect to the training accuracy transfers, the optimal learning rate with respect to the validation error is smaller for MLPs of moderate widths due to harmful overfitting. How to control for nonmonotonic dependence of the test error on the training error is an important question for future work. Figure VII.6 also shows μP-global but with a different choice of input and output multipliers. With these multipliers, networks with width at most 256 perform better in terms of test accuracy than with the other multiplier choice in Figure VII.5, but these multipliers have worse width scaling properties. To the best of our knowledge, the issue that optimally tuned hyperparameters on small models may scale worse than slightly suboptimal hyperparameters has not been stated before. This raises the question when and how can we use small models to predict the optimal choice of all hyperparameters jointly in large models.

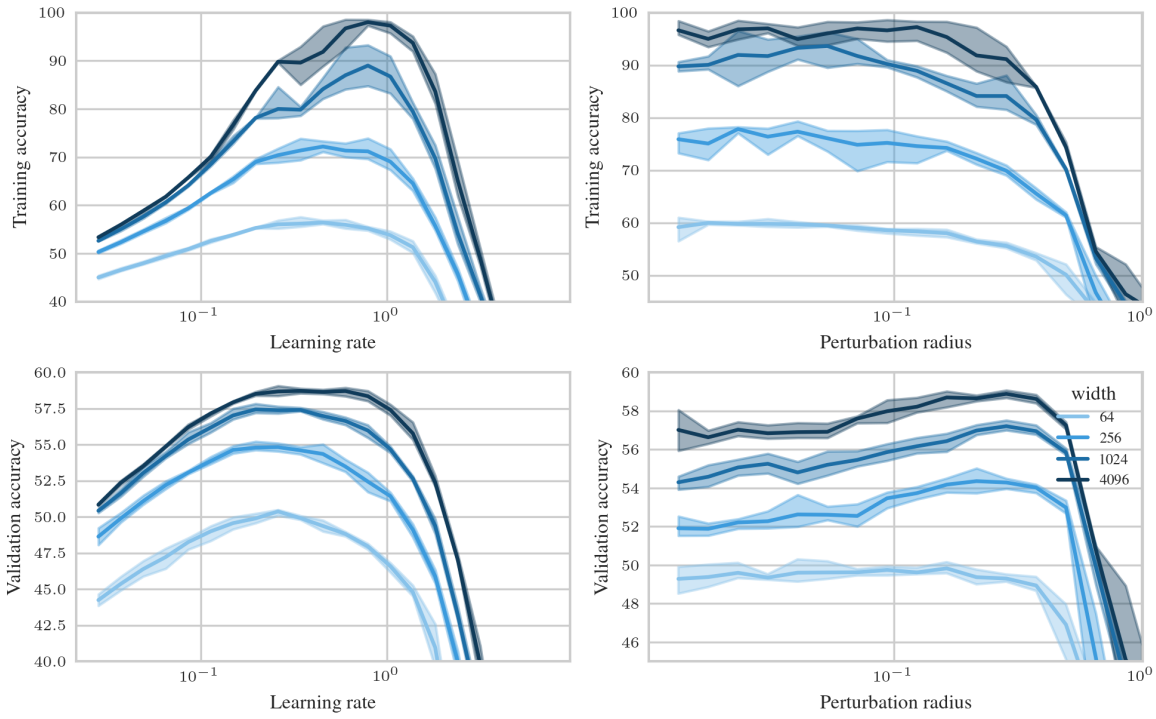


Figure VII.4: Training accuracy (top) and test accuracy (bottom) after optimally stopping 20 epoch SAM training as a function of learning rate (left) with perturbation radius $\rho = 0.2154$, and as a function of perturbation radius (right) with learning rate $\eta = 0.4529$ in μP^2. The optimal learning rate transfers. The smaller the perturbation radius the better the training accuracy. For sufficiently wide MLPs, the validation-optimal perturbation radius transfers as well and SAM reduces harmful overfitting.

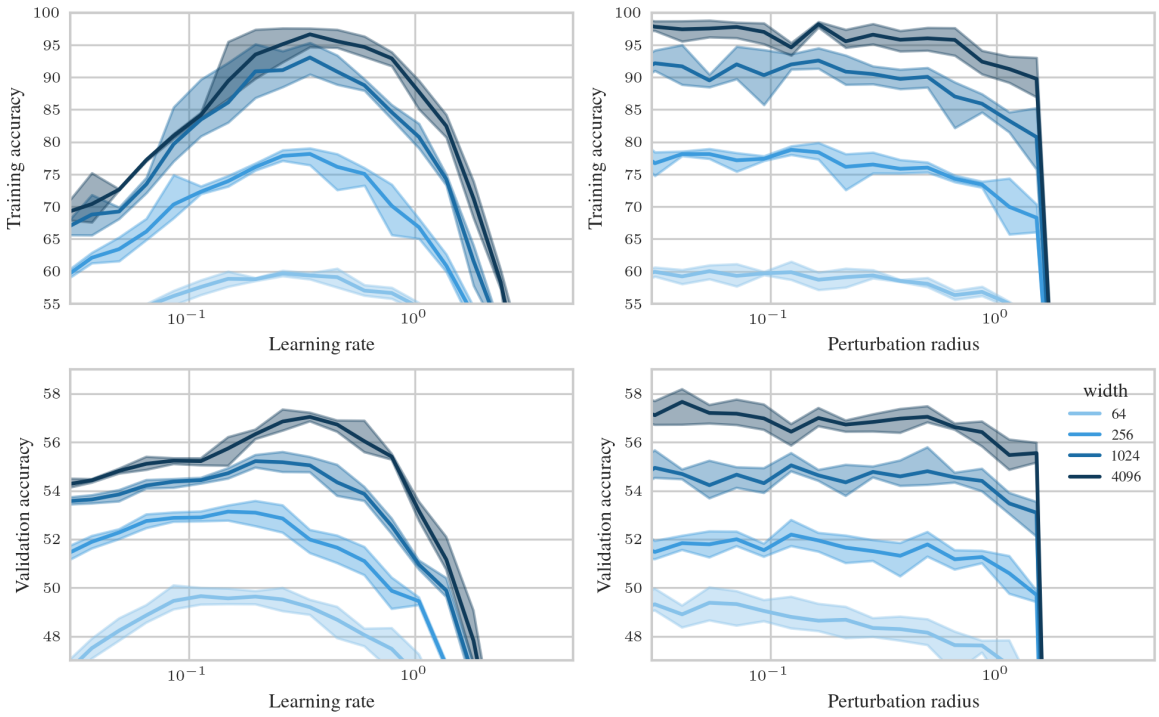


Figure VII.5: Training accuracy (top) and test accuracy (bottom) after optimally stopping 20 epoch SAM training as a function of learning rate (left) and perturbation radius (right) in μP-global with the same base learning rate and perturbation radius as in Appendix VII.3.1. For global perturbation scaling, we do not observe a benefit of SAM over SGD.

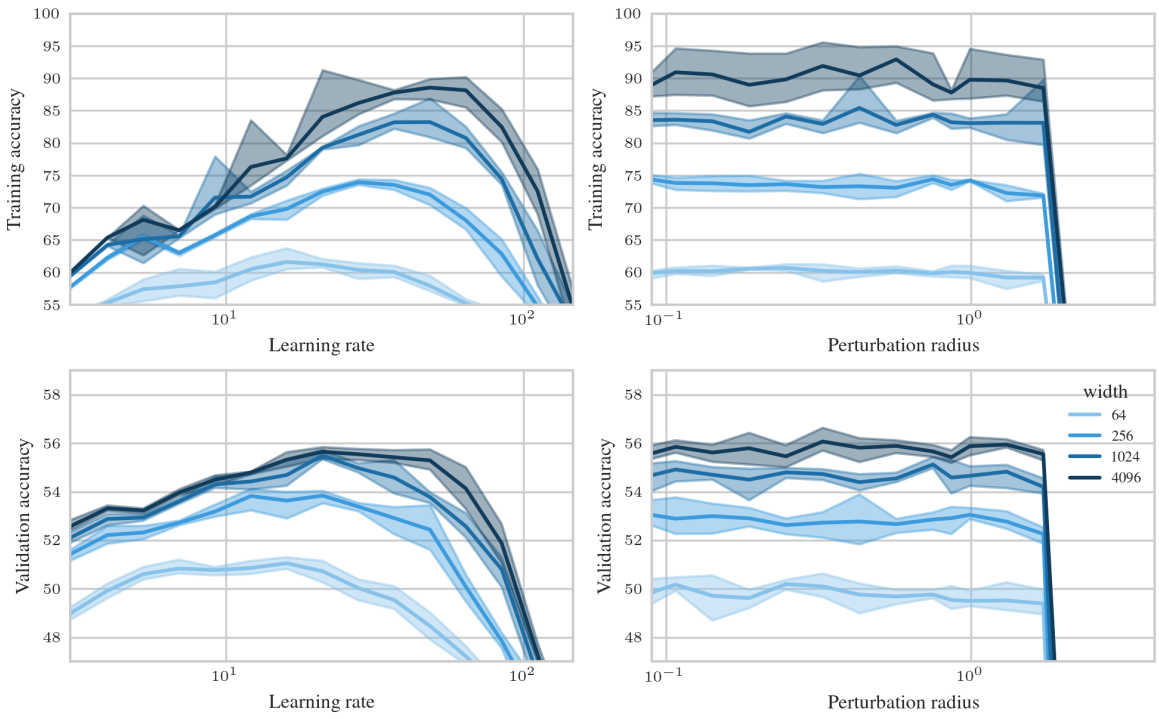
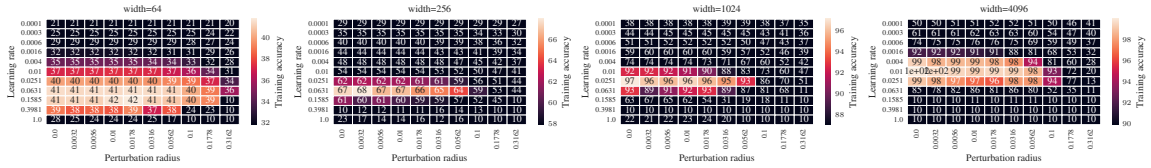
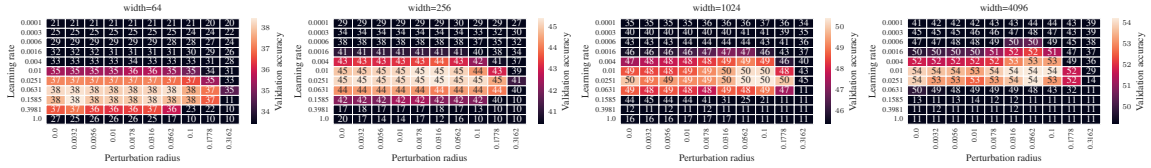


Figure VII.6: Same as Figure VII.5 but with input multiplier 0.0305 and small output multiplier 0.0098. Note that networks with width at most 256 perform better in terms of test accuracy than with the other multiplier choice in Figure VII.5, but the multipliers here have worse width scaling properties. To the best of our knowledge, the issue that optimally tuned hyperparameters on small models may scale worse than slightly suboptimal hyperparameters has not been stated before. This raises the question when and how can we use small models to predict the optimal hyperparameters of large models.

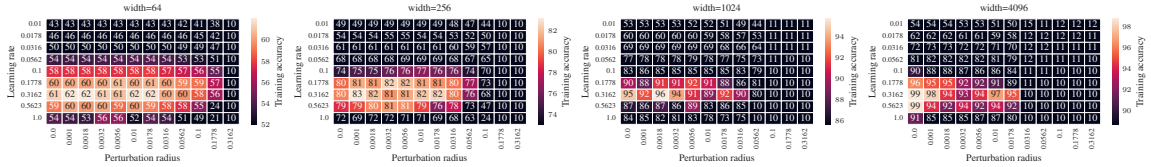


(*a*)

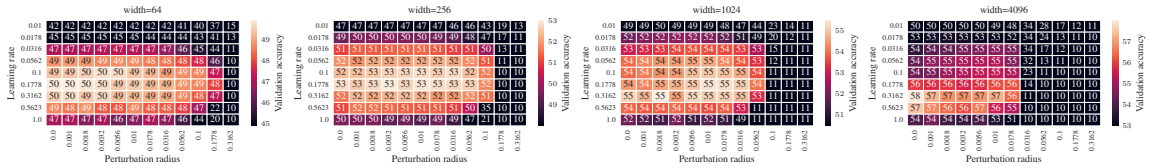


(*b*)

Mean (over 2 runs) of training accuracy (top) and test accuracy (bottom) after optimally stopping 20 epoch SAM training of a MLP in SP-naive as a function of learning rate and perturbation radius. Neither the optimal learning rate nor the optimal perturbation radius transfers. Every fixed learning rate becomes unstable in sufficiently wide networks. Optimal training and test accuracy are reached on differing hyperparameters due to harmful overfitting in SGD.

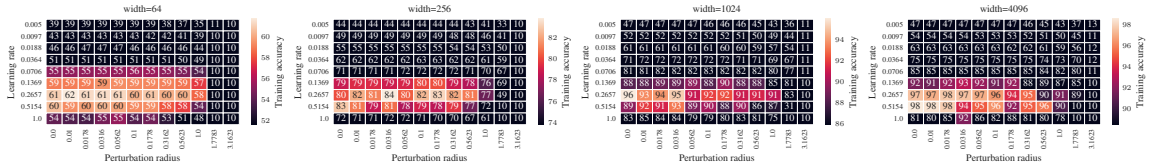


(*c*)

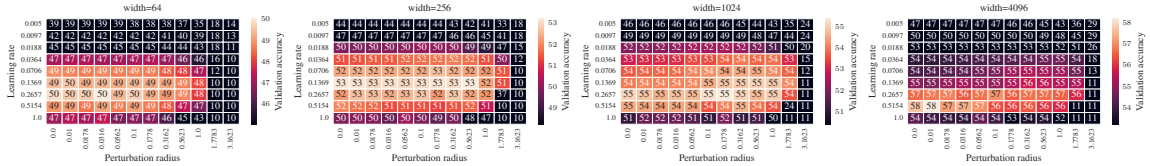


(*d*)

Mean (over 3 runs) of training accuracy (top) and of test accuracy (bottom) after optimally stopping 20 epoch SAM training of a MLP in μP-naive as a function of learning rate and perturbation radius. The optimal hyperparameters do not transfer. Every fixed perturbation radius becomes unstable in sufficiently wide networks.

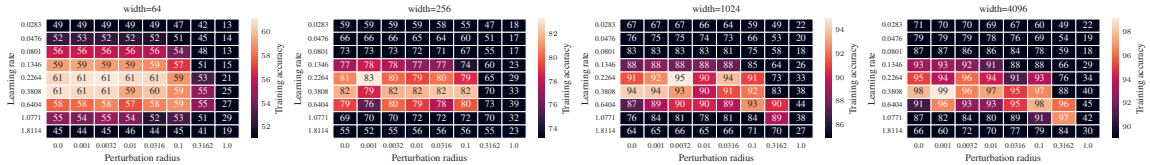


(*e*)

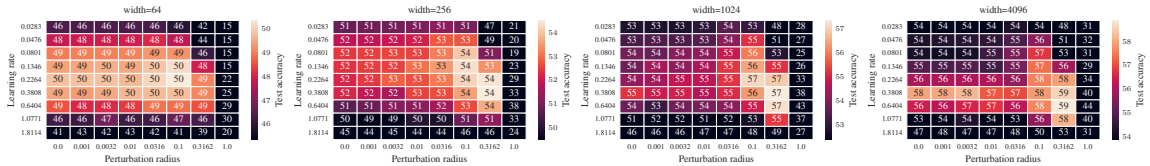


(*f*)

Mean (over 3 runs) of training accuracy (top) and of test accuracy (bottom) after optimally stopping 20 epoch SAM training of a MLP in μP-global as a function of learning rate and perturbation radius. The global scaling of the perturbation radius by $n^{-1/2}$ compared to μP-naive (Appendix VII.3.1) makes the stable regime invariant to width. But the suboptimal layerwise perturbation scaling that only perturbs the last layer does not consistently improve over SGD ($\rho = 0$).



(*g*)



(*h*)

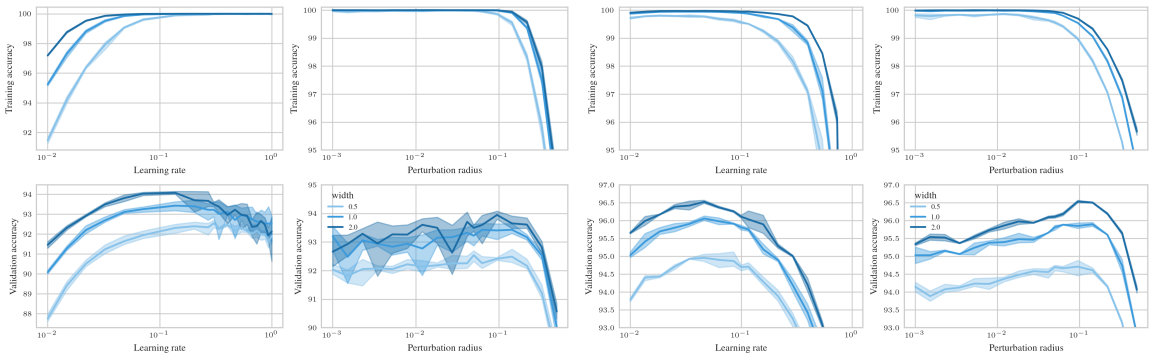
Mean (over 3 runs) of training accuracy (top) and of test accuracy (bottom) after optimally stopping 20 epoch SAM training of a MLP in μP^2 as a function of learning rate and perturbation radius. At sufficient width, the optimal hyperparameters are stable in terms of test accuracy, even under severe overfitting.

VII.3.2. RESNETS

In this section, we plot averages and σ-CI from 2 independent runs.

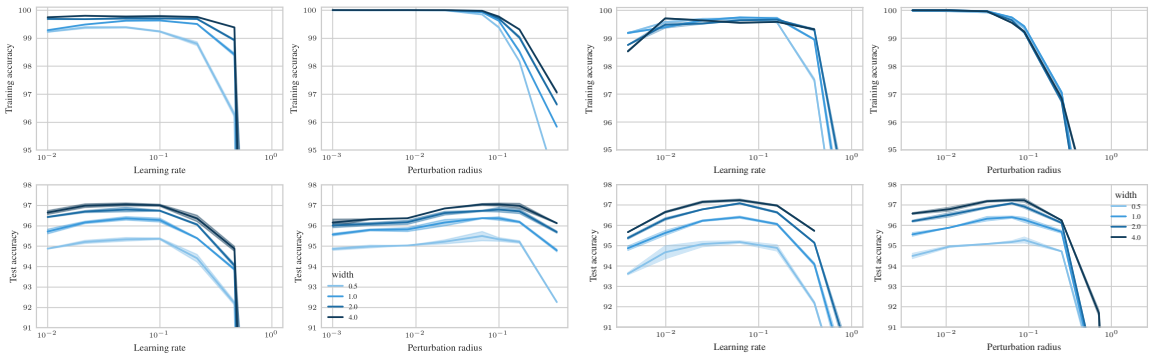
ResNets in μP^2 transfer both the optimal learning rate and perturbation radius for SAM (Appendix VII.3.2), SAM-ON (Appendix VII.3.3) and elementwise ASAM (Appendix VII.3.3), as well as different alternatives of scaling the gradient norm contributions to SAM's denominator (Appendix VII.4). This suggests correctness of the derived scalings. At width multipliers 2 and 4, μP^2 achieves the same or slightly better test accuracy than SP in all SAM variants.

Figure VII.7 shows ResNets trained with SAM in different parameterizations. In ResNets of practical scale, ρ remains quite stable in several parameterizations. In μP, for naive perturbation scaling the optimal perturbation radius shrinks, for global perturbation scaling, the optimal perturbation radius approaches is maximal stable value. μP^2 is most robust to the choice of ρ and achieves the best test accuracy. Surprisingly, ResNets in SP have very stable hyperparameter transfer across most SAM variants too, as soon as we tune momentum, weight decay and labelsmoothing (Appendix VII.3.2). This is in line with previous empirical observations (Figure 16, 54) but contradicts pure infinite-width theory. Because we are training to convergence, pure infinite-width theory does not adequately describe the training dynamics anymore [44]. We plan to study this phenomenon in more detail in upcoming work. The infinite-width theory implies that scaling the width further would eventually break the learning rate transfer.



(*i*) (*j*)

Training accuracy (top) and test accuracy (bottom) after optimally stopping 100 epoch SAM training as a function of learning rate and perturbation radius in SP-naive without regularization (left) and with tuned regularization (right) using momentum 0.9, weightdecay 0.0005 and labelsmoothing 0.1. CI denote the minimal and maximal value from 4 independent runs. Without regularization, the optimal learning rate shrinks with width. Given the learning rate, the optimal perturbation radius seems quite stable, but since the optimal learning rate shifts, the performance scales worse than for μP^2 with the fixed learning rate that is tuned on the small model. With optimal regularization, both optimal learning rate and perturbation radius remain remarkably stable. We plan to investigate this mechanism in an upcoming work.



(*k*) (*l*)

Training accuracy (top) and test accuracy (bottom) after optimally stopping 200 epoch SAM training as a function of learning rate and of perturbation radius in SP (left) and μP^2 (right) with optimized momentum 0.9, weight decay $5 \cdot 10^{-4}$ and labelsmoothing 0.1 for both μP^2 and SP. In μP^2, the base learning rate is $\eta = 2^{-4}$ and the base perturbation radius is $\rho = 2^{-4}$, in SP $\eta = 0.05$ and $\rho = 0.1$, respectively. Observe monotonic improvement with width in both training and test error. Optimal hyperparameters transfer across widths, surprisingly in both μP^2 and SP.

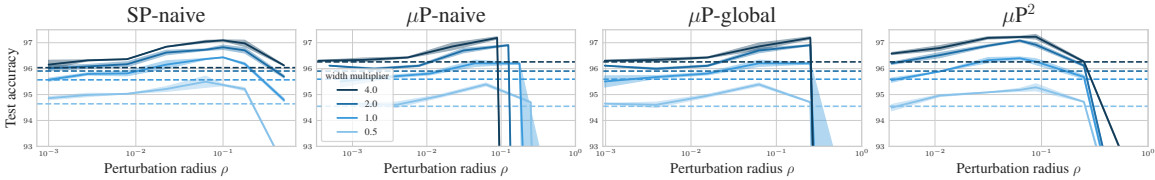


Figure VII.7: Test accuracy after optimally stopping 200 epoch SAM training as a function of perturbation radius in various parameterizations. Dashed lines denote the base optimizer SGD with tuned momentum and weight decay in the respective parameterization.

	SAM global	SAM μP^2	SAM-ON μP^2	Elem. ASAM μP^2
SP	$97.00_{\pm 0.03}(+0.96)$	$97.00_{\pm 0.03}(+0.96)$	$\mathbf{97.29}_{\pm 0.06}(\mathbf{+1.26})$	$97.15_{\pm 0.01}(+1.11)$
μP	$\mathbf{97.19}_{\pm 0.05}(\mathbf{+0.93})$	$\mathbf{97.23}_{\pm 0.08}(\mathbf{+0.97})$	$\mathbf{97.34}_{\pm 0.08}(\mathbf{+1.08})$	$\mathbf{97.32}_{\pm 0.05}(\mathbf{+1.06})$

Table VII.1: **(Performance of μP^2)** Average test accuracy$_{\pm \text{standard deviation across 4 runs}}$ (+ improvement of SAM over SGD) for ResNet-18 with width multiplier 4 on CIFAR10 using SGD as a base optimizer. In bold, all parameterizations within a 2σ-CI from the best-performing variant SAM-ON in μP^2.

VII.3.3. ASAM VARIANTS

As we are not aware of any use of ASAM with MLPs in the literature and since the amount of necessary experiments for ViTs exceeds our computational budget, we only show that ResNets trained with the all of the discussed SAM variants in μP^2 transfer the optimal (η, ρ).

Figure VII.8 shows that training a ResNet-18 in μP^2 achieves hyperparameter transfer in ρ for most considered SAM variants with varying width. μP with global perturbation scaling (μP-global) becomes unstable if ρ is chosen slightly larger than optimal. Table VII.1 shows that all SAM variants perform similarly well in μP^2, some slightly outperforming the best-performing variant SAM-ON in SP. This suggests that for ResNets, even with a proper layerwise balance, normalization layer perturbations may suffice, and performance differences in SP are primarily caused by varying degrees to which the normalization layers are perturbed. Our results partially explain the success of SAM-ON, as all normalization layers are already effectively perturbed without layerwise perturbations (see Appendix V.5). For transferring the optimal ρ with SAM-ON in μP, our theory predicts the global scaling $\rho = \Theta(n^{1/2})$ which is confirmed by our empirical observations (Figure VII.8). However, properly understanding the role of normalization layer perturbations remains an important question for future work. Note that we report results after fine-tuning all hyperparameters. The performance gain of μP^2 over SP is likely much higher in larger models, for which fine-tuning is infeasible and the lack of feature learning and effective perturbations is more pronounced. Even under optimal HPs, μP^2 appears to stabilize SAM's training dynamics compared to SP (**??**).

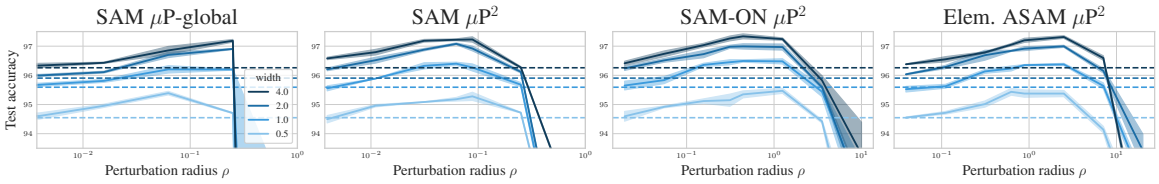
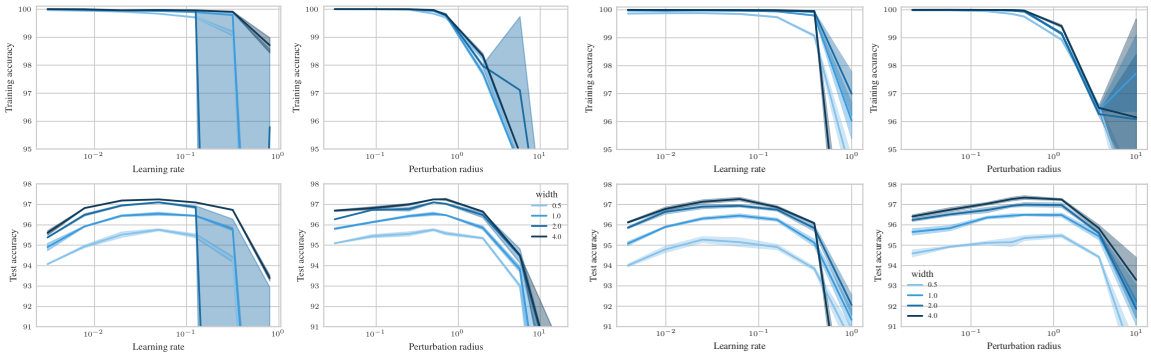


Figure VII.8: **(ρ-transfer of ASAM variants in μP^2)** Test error as a function of perturbation radius ρ after 200 epochs of training a ResNet-18 in μP^2 on CIFAR10 with various SAM variants (see subplot title). CI over 2 independent runs. Darker lines correspond to larger width multipliers. Other hyperparameters are tuned at base width multiplier 0.5. μP^2 achieves transfer in ρ and large improvements over the base optimizer (dashed lines) SGD in μP with momentum and weight decay.

For the examples of elementwise ASAM and SAM-ON the global perturbation scaling $n^{1/2}$ suffices to reach μP^2. The stability of the optimal perturbation radius in the applied scaling $n^{1/2}$ shows that in μP with naive perturbation scaling the optimal perturbation radius would grow as $n^{1/2}$.

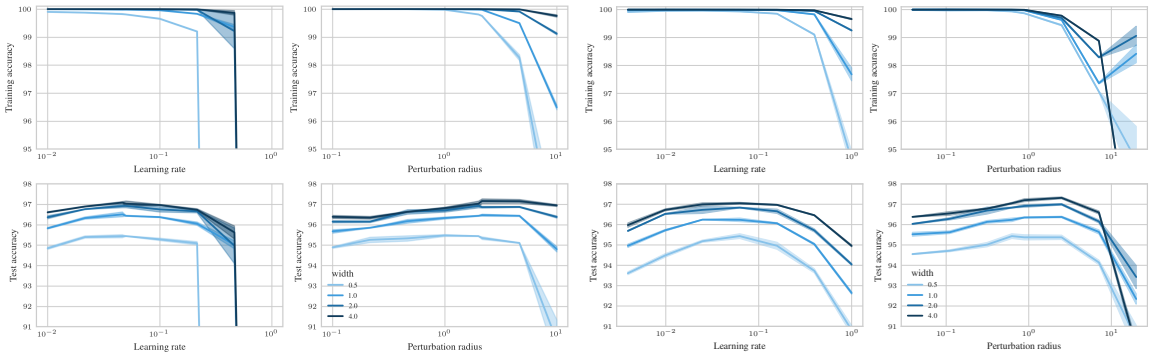
See the previous section, for a discussion of the remarkable stability of ResNets in SP. For the example of elementwise ASAM in SP, the optimal perturbation radius seems to grow.

For layerwise ASAM (Appendix VII.3.3), the optimal perturbation radius seems to grow in both SP and μP^2, suggesting that our scaling condition does not perfectly apply to this variant, although μP^2 ($97.09_{\pm 0.03}(+0.83)$) still outperforms SP ($96.86_{\pm 0.05}(+0.83)$) in terms of the optimal test accuracy. As Frobenius norms of weights are the only component that is not representable as a NE$\otimes$OR$\top$program, these Frobenius norms appear to scale differently than heuristically predicted over the course of training.



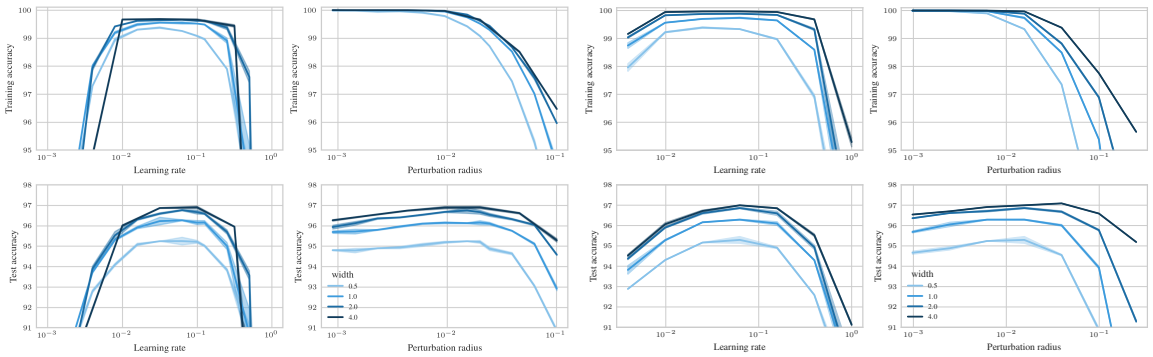
(*a*) (*b*)

Same as Appendix VII.3.2 but for SAM-ON in μP^2 (left) and SP without perturbation scaling (right). Both optimal learning rate and perturbation radius are remarkably stable in both μP^2 and SP. Since μP^2 for SAM-ON is just μP with global perturbation scaling $n^{1/2}$, transfer here implies that μP with width-independent scaling would not transfer.



(*c*) (*d*)

Same as Appendix VII.3.2 but for elementwise ASAM in SP without perturbation scaling (left) and in μP^2 (right). Observe a consistent HP landscape in μP^2 but growing optimal perturbation radius in SP without perturbation scaling.

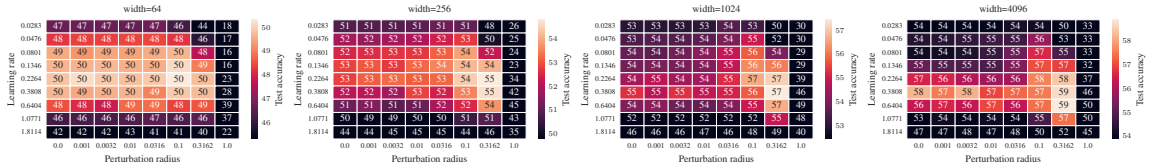


(*e*) (*f*)

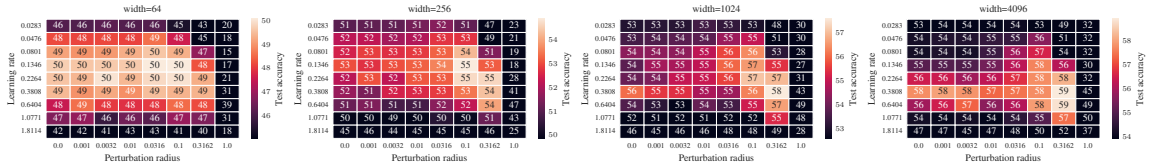
Same as Appendix VII.3.2 but for layerwise ASAM in SP without perturbation scaling (left) and μP^2 (right). For layerwise ASAM, both μP^2 and SP seem to transfer the optimal learning rate as well as perturbation radius.

VII.4. Gradient norm contributions have negligible effects on generalization performance

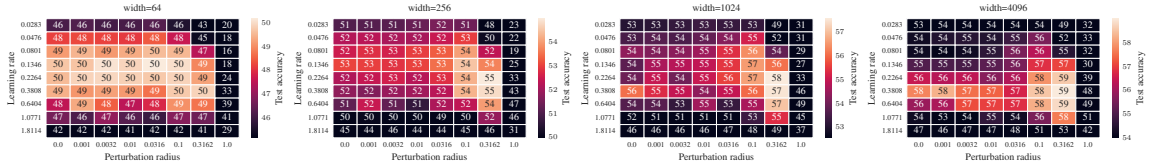
In this section we provide ablations concerning the question which layers should contribute non-vanishingly to the gradient norm in the denominator of the layerwise SAM perturbation rule (LP).



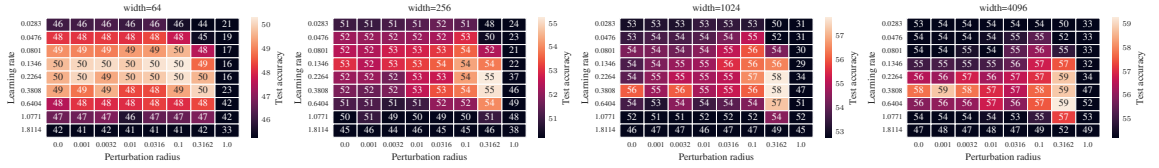
(*g*)



(*h*)



(*i*)



(*j*)

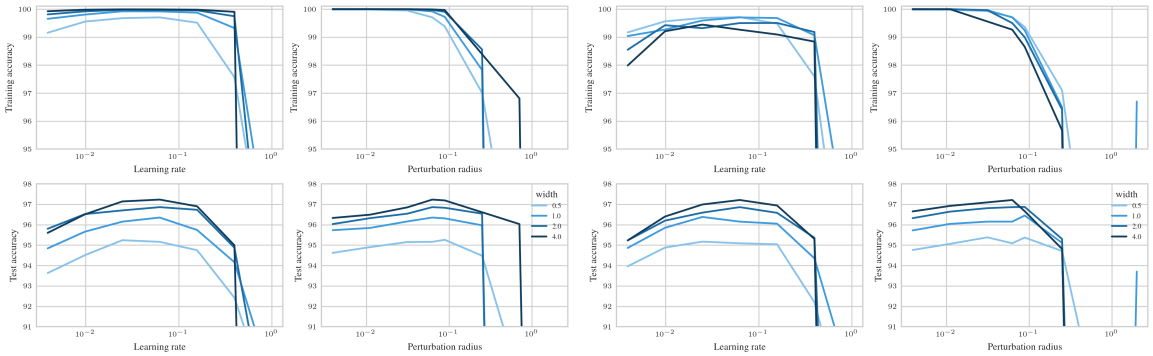
Scaling the gradient norm contributions of all layers to $\Theta(1)$ (first row) and then setting the first layer gradient norm to 0 (2nd row), respectively the hidden layer (3rd row), last-layer (4th row). Each individual layer seems to have vanishing contribution to the optimal test error.

While under the definition of *bcd*-parameterizations only the first layer contributes non-vanishingly to the gradient norm $\|v_t\|$ (Theorem 25), one could also decouple the numerator from the denominator scalings and scale the gradient norm contributions of all layers to $\Theta(1)$. Here, we find that this has a negligible effect on the optimal generalization performance, but can be more stable given slightly suboptimal hyperparameters.

For MLPs, in Appendix VII.4 we scale all contributions to $\Theta(1)$, and then set the contribution of individual layers to zero, one by one. We observe no significant effect on the optimal test loss or hyperparameter transfer for MLPs. Any layer's contribution to the gradient normalization in the denominator of the SAM update rule can be set to 0 without a significant effect on the test loss. This raises the question which effect the gradient normalization has in μP. Does it contribute a scaling correction in SP, but may be dropped entirely in μP?

For ResNets, Appendix VII.4(a) shows accuracies when rescaling all layers' gradient norms to $\Theta(1)$, and Appendix VII.4(b) shows the results when using the original global gradient norm rescaled to $\Theta(1)$. Again, both methods achieve similar optimal test accuracy. The first variant shows cleaner hyperparameter transfer and monotonous improvement with width. When comparing to our original definition (LP) in Appendix VII.3.2, optimal performance is similar but rescaling all layers' gradient

norm contributions to $\Theta(1)$ may even produce a slightly more stable hyperparameter-loss landscape for ResNets.



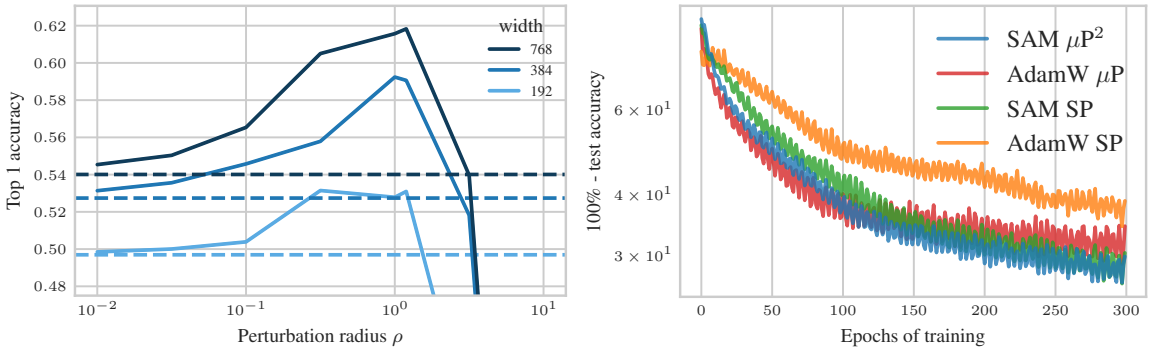
(*k*) (*l*)

Same as Appendix VII.3.2 but with scaling of the gradient norms in the SAM perturbation (LP) denominator that scales all terms to $\Theta(1)$ (left) and only global denominator scaling $\frac{\|\nabla L\|}{n_L}$ (right). All denominator scalings achieve similar optimal accuracy, show HP transfer in learning rate and monotonic test accuracy improvement with width. Only global denominator scaling does not have monotonically improving training error, and is slightly shifting at large widths.

VII.5. Vision Transformers in μP^2

Appendix VII.5(a) shows that μP^2 also achieves ρ-transfer and large improvements over the base optimizer AdamW in μP in ViTs [14] trained on Imagenet1K [12]. While Andriushchenko and Flammarion [1, Appendix E.3] observe diminishing benefits of SAM at large widths in SP, here the improvements beyond the base optimizer AdamW in μP are particularly large ($+8\%$ top 1 accuracy!).

Appendix VII.5(b) shows ViTs on CIFAR100 over the course of training. In ViTs, μP generally achieves decent accuracy faster than SP. Our theory suggests that in μP^2 the gradients are scaled correctly from the beginning, whereas in SP they have to self-stabilize first, which slows down convergence. We plan a closer analysis in an upcoming work. SAM converges slower than the base optimizer AdamW in favor of drifting towards a better generalizing local minimum or saddle point. For ViTs at this moderate scale, SAM in SP catches up to SAM in μP^2 at the end of training.



(m) (n)

(a) Training a ViT with SAM in μP^2 on ImageNet1K from scratch for 100 epochs yields ρ-transfer and large improvements over AdamW in μP (dashed lines). *(b)* Training a ViT with width multiplier 2 on CIFAR100 with AdamW as the base optimizer.