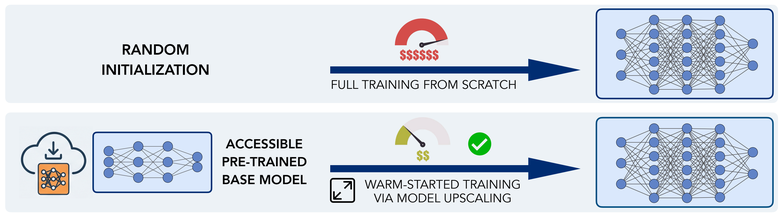
μpscaling Small Models
Principled Warm Starts and Hyperparameter Transfer

Yuxin Ma Nan Chen Mateo Díaz Soufiane Hayou Dmitriy Kunisky Soledad Villar
Department of Applied Mathematics and Statistics, Johns Hopkins University

ICML

arXiv

Model Upscaling



Prior work & limitations

- Based on function-preserving weight expansion [1].
- **The critical gap:** Performance is sensitive to hyperparameters, and tuning at the target scale is expensive.

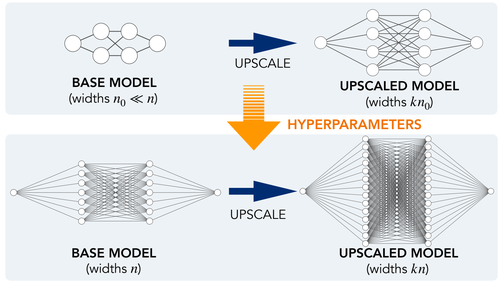
Our contribution:
a **theoretically-grounded** upscaling method (with respect to widths) that allows **hyperparameter transfer** via μP.

Proposed Upscaling Algorithm

Meta-algorithm 1 Upscaling procedure

Input: Base model checkpoint at width n with the optimizer checkpoint; expansion factor k.
Output: Trained upscaled model at width $N = nk$.

0. (HP tuning) Tune $\overline{\sigma_\Delta}$ and $\overline{\gamma^\uparrow}$ by running Steps 1-4 on a small proxy ($n_0 \to kn_0$, $n_0 \ll n$) and selecting values that minimize terminal training loss.
1. Construct an equivalent wider model of width N by duplicating and rescaling the weights.
2. Add noise to the weights following the μP initialization scaling, controlled by $\overline{\sigma}_\Delta$.
3. Transfer the base optimizer's internal states to the upscaled model's optimizer, analogously to the weights.
4. Train the upscaled model under μP with LR base constant $\overline{\gamma^\uparrow}$.



Maximal Update Parametrization (μP)

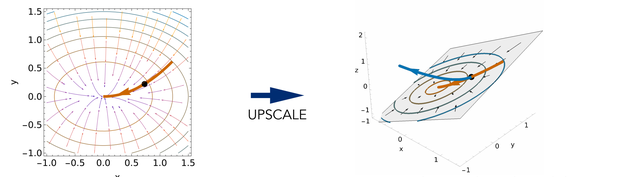
Weight type	Init Var. $\mathcal{N}(0, B\cdot\overline{\sigma}^2)$	LR $\gamma = C\cdot\overline{\gamma}$	WD (vanilla) $\lambda = D\cdot\overline{\lambda}$	WD (decoupled) $\lambda = \hat{D}\cdot\overline{\lambda}$	Extra HPs $\varepsilon = E\cdot\overline{\varepsilon}$	Output multiplier
Vector-like	1	n^m	n^{-1}	n^{-m}	n^{-1}	n^{-1}
Matrix-like	n_{in}^{-1}	$n_{\text{out}}^m n_{\text{in}}^{-1}$	$n_{\text{in}} n_{\text{out}}^{-1}$	$n_{\text{in}} n_{\text{out}}^{-m}$	n_{out}^{-1}	$--$

- m depends on the optimizer: $m = 1$ for SGD and $m = 0$ for Adam.
- ε stands for extra hyperparameters that require rescaling, such as the numerical stability constant `eps` in Adam.

- Yields optimal feature learning behavior in the infinite-width limit [2].
- Allows zero-shot hyperparameter transfer for training from scratch [3].

From Static Equivalence to Dynamic Equivalence

Theoretical guarantee 1:
Injecting zero noise ($\overline{\sigma_\Delta} = 0$) yields a widened model whose training trajectory **exactly** matches that of the base model.



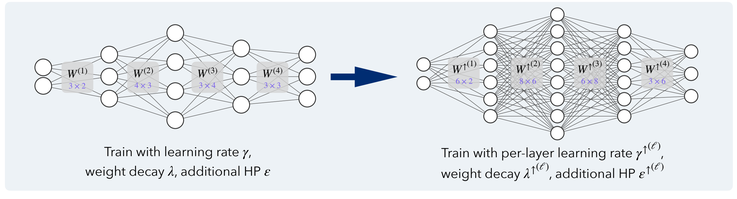
→ training trajectory of the base model

→ equivalent widened model
→ upscaled model with added noise

Theorem: For standard architectures (e.g., MLP, CNN, RNN, Transformer) and optimizers (e.g., SGD, Adam), the following models are ***dynamically equivalent*** (expresses the same function at each training step).

- The narrow **base model** trained with given HPs.
- The **widened model** (weights constructed in the following way) trained under μP with the same HP base constant as the base model.

Vector-like	$n \times d \to kn \times d$	`W.repeat_interleave(k, dim=0)`
Matrix-like	$n_{\text{out}} \times n_{\text{in}} \to k_{\text{out}} n_{\text{out}} \times k_{\text{in}} n_{\text{in}}$	`W.repeat_interleave(k_out, dim=0)` `.repeat_interleave(k_in, dim=1) / k_in`



Infinite-Width Limit and Hyperparameter Transfer

- μP-based HP transfer [3] applies to training from scratch only; the Tensor Programs framework [2,3] does not support the widening operation.
- We extend Tensor Programs to support mid-training upscaling.
- **Key idea:** Partition the width-kn activations into k groups of n neurons each. Each group uses only standard operations, placing upscaled training within the existing framework. In the infinite-width limit, activations now behave like i.i.d. k-dim Gaussian blocks with non-trivial covariance, rather than i.i.d. scalars.
- **Result:** Our algorithm's design choices are precisely those that keep the entire training process (including mid-training upscaling) under μP.

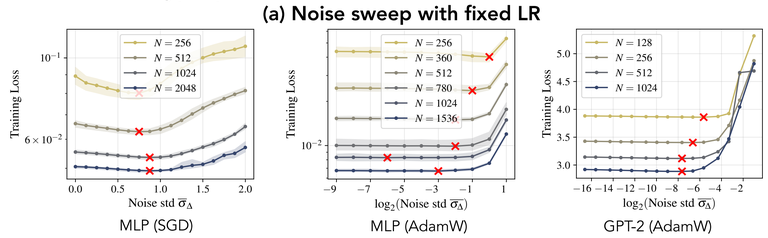
Theoretical guarantee 2:
In the infinite-width limit, the injected noise is both **safe** (base-model signal preserved) and **useful** (noise's contribution neither vanishes nor explodes).

Theoretical guarantee 3:
Hyperparameters $\overline{\sigma}_\Delta$ (noise level) and $\overline{\gamma^\uparrow}$ (upscaled model LR) can be tuned efficiently via μP-grounded **hyperparameter transfer**.

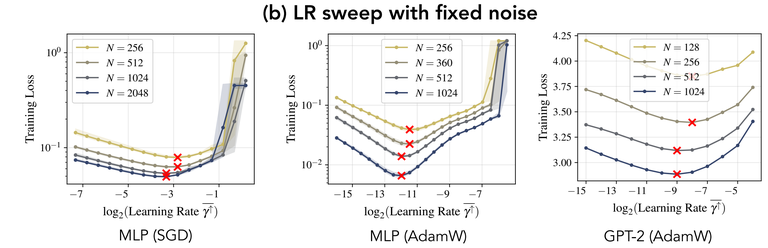
Experiments

Validation of hyperparameter transfer

(a) Noise sweep with fixed LR

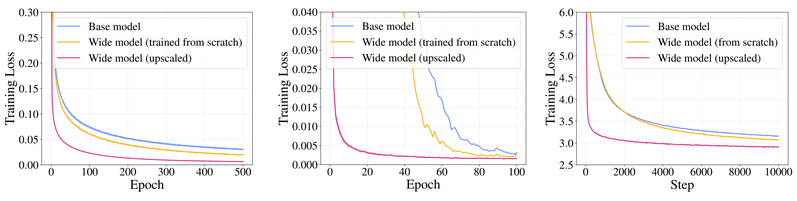


(b) LR sweep with fixed noise



Upscaling accelerates training

MLP (AdamW) on Forest Cover Type	ResNet (SGD) on CIFAR-100	GPT-2 (AdamW) on FineWeb
width $500 \to 2000$	$2\times \to 4\times$ model	width $1920 \to 3840$
tuning on width $100 \to 400$	tuning on $0.5\times \to 1\times$ model	tuning on width $192 \to 384$
$23.6\times$ tuning-FLOP reduction	$16.0\times$ tuning-FLOP reduction	$48.2\times$ tuning-FLOP reduction



Conclusions and Future Work

Key contributions

- Provides the first rigorous HP transfer framework for model upscaling, applicable to general architectures and optimizers.
- Extends the theoretical foundation from static to dynamic equivalence.
- Connects model upscaling to the μP infinite-width theory.

Future work

- Extend to depth-wise and joint width-and-depth upscaling.
- Understand generalization behavior under upscaling (current framework addresses training dynamics only).
- Validate at industry-scale settings, where larger target-to-tuning ratio n/n_0 yields greater HP tuning cost savings.

References

[1] Chen, Tianqi, Ian Goodfellow, and Jonathon Shlens. **"Net2Net: Accelerating Learning via Knowledge Transfer."** *ICLR (2016).*
[2] Yang, Greg, and Edward J. Hu. **"Tensor programs IV: Feature learning in infinite-width neural networks."** *ICML* (2021).
[3] Yang, Greg, et al. **"Tensor programs V: Tuning large neural networks via zero-shot hyperparameter transfer."** *NeurIPS (2021).*