Scaling neural networks

Cengiz (“Jen-ghiz”) Pehlevan

Harvard John A. Paulson
School of Engineering
and Applied Sciences

Kempner
INSTITUTE

For the Study of Natural
& Artificial Intelligence
at Harvard University

AI is changing the world!

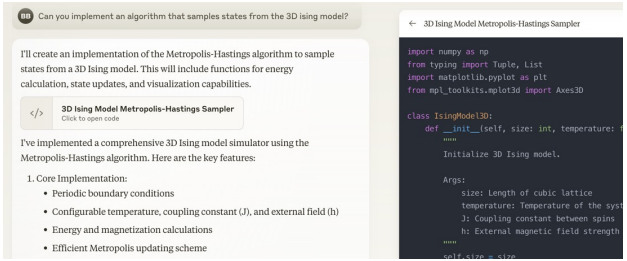
Language Models for Text and Code Generation

ChatGPT

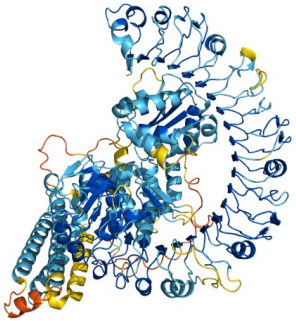
Claude

Gemini

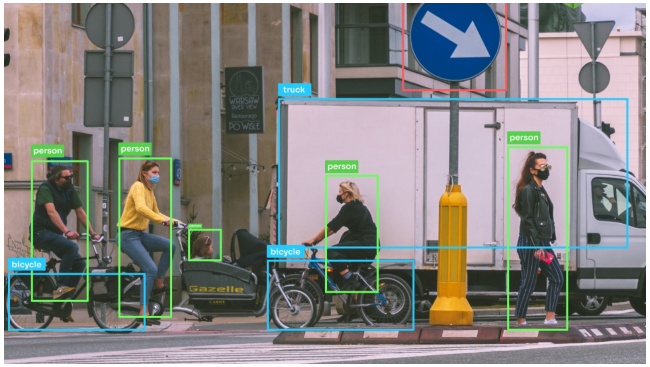
LLaMA by Meta



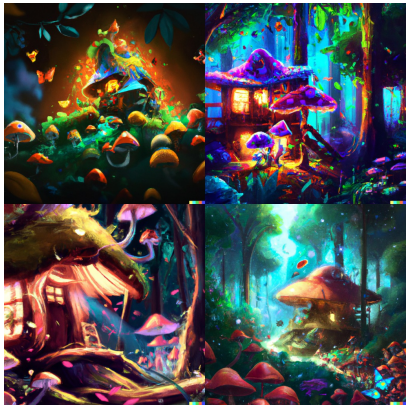
Biology (Protein Folding)



Vision Models (Object Recognition)



(Generative Image/Video Models)



Why is this happening?

Dominant AI Paradigm: Scaling

Hestness et al, 2017, “Deep Learning Scaling is Predictable, Empirically”

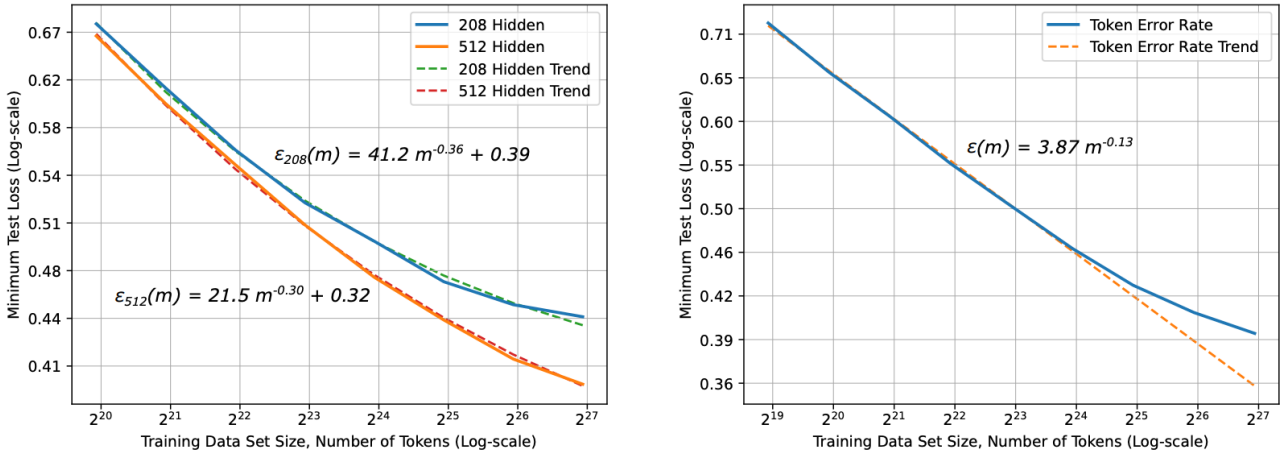
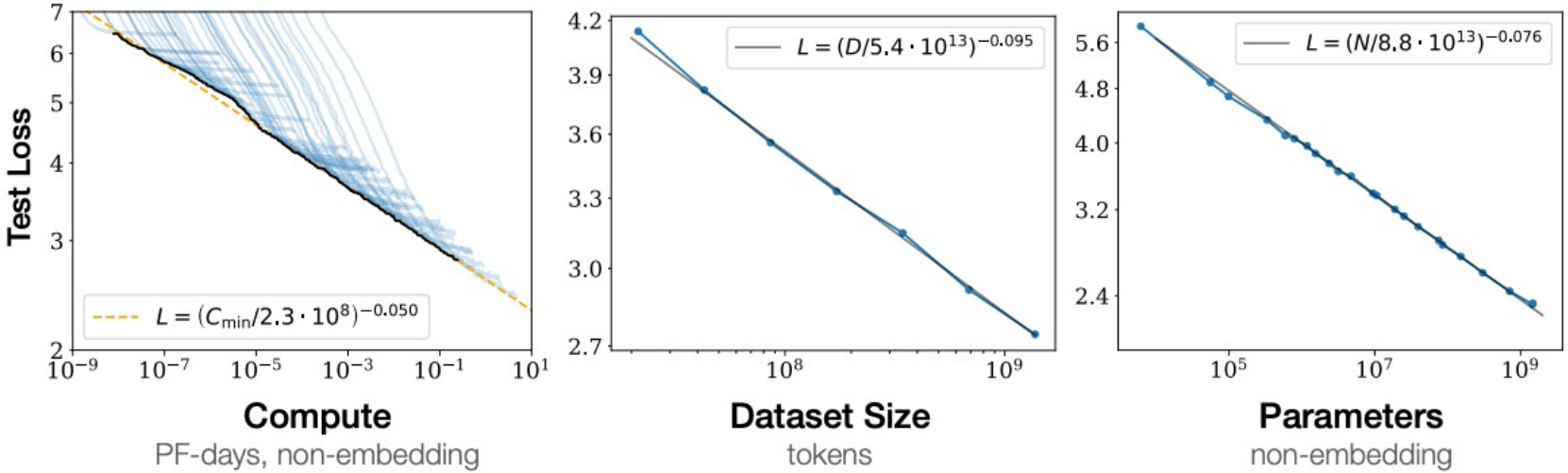


Figure 1: Neural machine translation learning curves. Left: the learning curves for separate models follow $\varepsilon(m) = \alpha m^{\beta_g} + \gamma$. Right: composite learning curve of best-fit model at each data set size.

Dominant AI Paradigm: Scaling

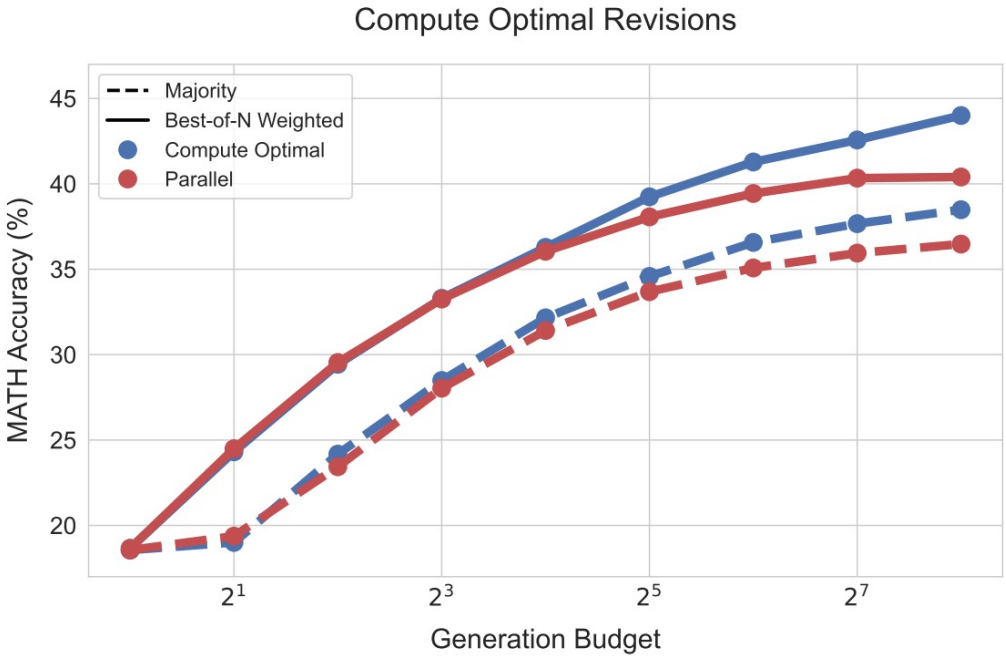
Kaplan et al 2020, “Scaling Laws for Neural Language Models”



Following these trends, 10x of compute leads to 10% reduction in loss

Test time scaling

Snell et al 2024, “Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters”



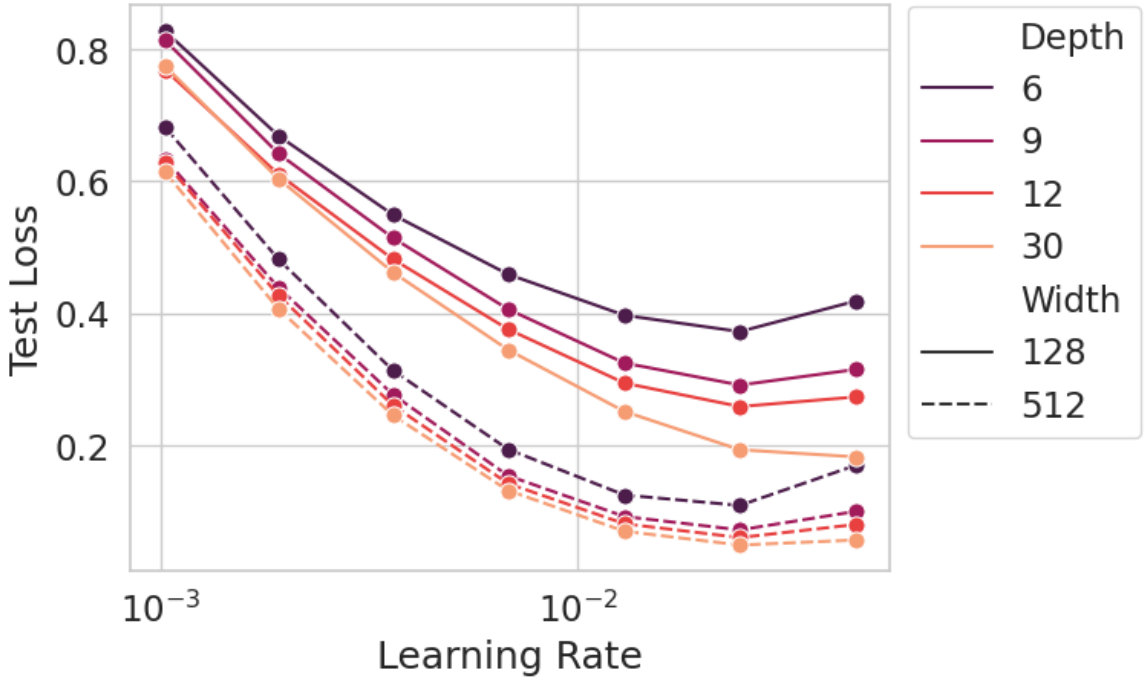
GPT-4 Technical Report

OpenAI*

3 Predictable Scaling

A large focus of the GPT-4 project was building a deep learning stack that scales predictably. The primary reason is that for very large training runs like GPT-4, it is not feasible to do extensive model-specific tuning. To address this, we developed infrastructure and optimization methods that have very predictable behavior across multiple scales. These improvements allowed us to reliably predict some aspects of the performance of GPT-4 from smaller models trained using $1,000\times$ – $10,000\times$ less compute.

Hyperparameter Transfer



ResNet, CIFAR-10

Yang et al., 2021; Bordelon et al., 2023;

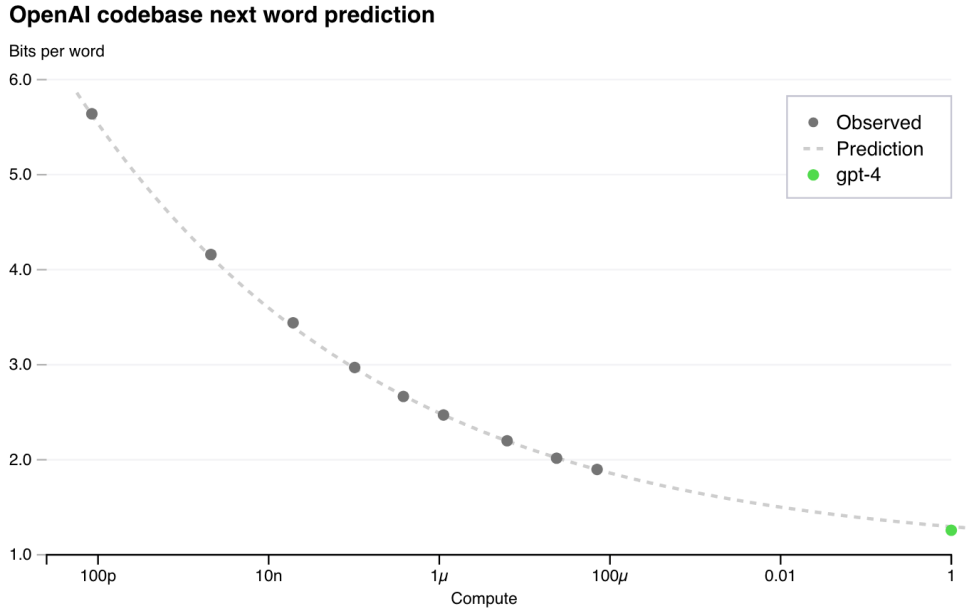


Figure 1. Performance of GPT-4 and smaller models. The metric is final loss on a dataset derived from our internal codebase. This is a convenient, large dataset of code tokens which is not contained in the training set. We chose to look at loss because it tends to be less noisy than other measures across different amounts of training compute. A power law fit to the smaller models (excluding GPT-4) is shown as the dotted line; this fit accurately predicts GPT-4's final loss. The x-axis is training compute normalized so that GPT-4 is 1.

Caution: not everything is predictable from small scale

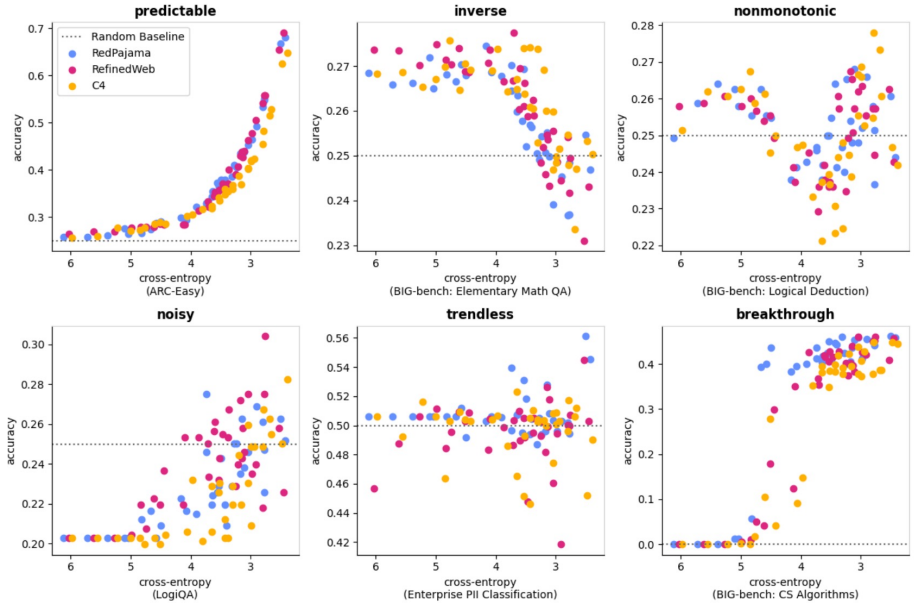
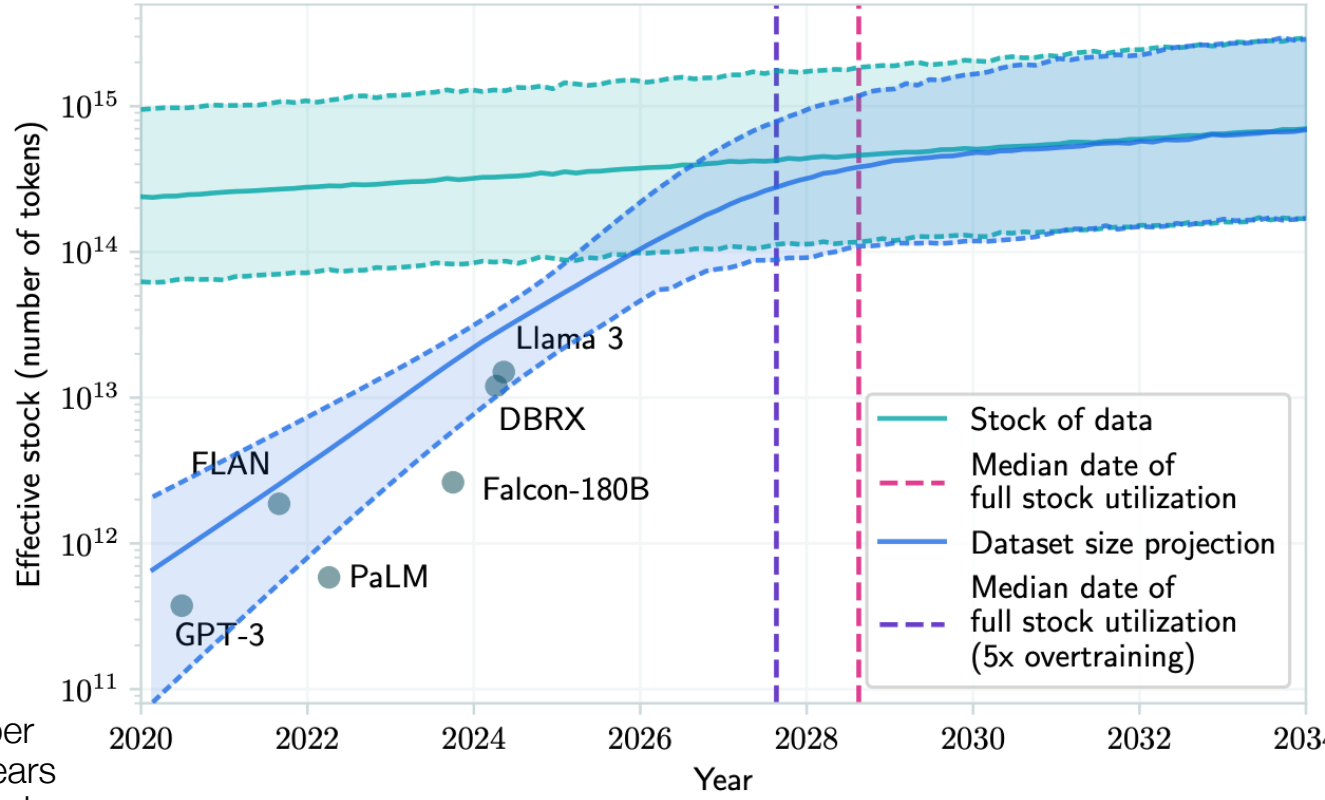


Figure 2: A taxonomy of different scaling behaviors. Predictable scaling fits closely to a linear functional form after, for example, exponentiating the cross-entropy loss. However, depending on the downstream task, models do not always improve with scale (inverse, nonmonotonic, and trendless), or the improvement might be highly noisy. The improvement might also follow a functional form that is difficult to extrapolate like a sigmoid (breakthrough).

Figures from Lourie, Hu and Cho, 2025

Challenges of Scaling Paradigm

Expensive: Current approaches are very data and compute hungry



100 pages per day for 100 years < 2 billion words

Villalobos et al 2024

Energy: According to internet rumor and some reputable sources (World Economic Forum), training GPT-4 consumed an estimated 50 gigawatt-hours (GWh) of energy, took 100 days and cost $100M USD! Decent part of the cost is due to performing large number of training runs.

ChatGPT uses ~0.5 GWh per day! (Forbes)

Compare to typical US household ~30 kWh per day!

FT Financial Times

AI set to fuel surge in new US gas power plants

The US is on the cusp of a natural gas power plant construction boom, as Big Tech turns to fossil fuels to meet the huge electricity needs...

Georgia Institute of Technology

AI's Energy Demands Spark Nuclear Revival

The demand for electricity to power AI data centers is skyrocketing, placing immense pressure on traditional energy sources.

Many interesting questions

- What are the limits of scaling?
    - “Universality classes”
    - What properties scale predictably, what do not?
- When does consistent behavior arise? Hyperparameter transfer
- What sets these scaling laws? Why power laws?
- Can we beat them? Better scaling laws.
- At what scale do new capabilities emerge? Important implications for AI Safety
- Is scaling sufficient? Does GPT-5 signal the end of scaling era?
- …..

In this series of talks

- Parameterizations for predictable scaling: How to scale up a neural network such that they converge to well-behaved limits and allow hyperparameter transfer?

- An introduction to Dynamical Mean Field Theory (DMFT) descriptions of infinite limits: We will discuss deeper and other architectures that will lead to a different description than presented in Andrea's talk

- Application of DMFT for understanding scaling laws

- A toy model of emergence of in context learning

Resources

LECTURE NOTES, REVIEWS AND BLOG POSTS

Replica Method for the Machine Learning Theorist - Part 1, by Blake Bordelon, Haozhe Shan, Abdul Canatar, Boaz Barak, Cengiz Pehlevan

Replica Method for the Machine Learning Theorist - Part 2, by Blake Bordelon, Haozhe Shan, Abdul Canatar, Boaz Barak, Cengiz Pehlevan

Lecture notes on the replica method for Wishart matrix eigenvalues, by Jacob Zavatone-Veth

A brief introduction to the neural network Gaussian process from the perspective of mean field theory, by Jacob Zavatone-Veth

Lecture Notes on Infinite-Width Limits of Neural Networks, Cengiz Pehlevan and Blake Bordelon, prepared for 2023 Princeton ML Theory Summer School

Infinite Limits of Neural Networks, Deeper Learning Blog, by Alex Atanasov, Blake Bordelon and Cengiz Pehlevan

A Dynamical Model of Neural Scaling Laws, Deeper Learning Blog, by Blake Bordelon, Alex Atanasov and Cengiz Pehlevan

Scaling and renormalization in high-dimensional regression, by Alex Atanasov, Jacob Zavatone-Veth, Cengiz Pehlevan

Solvable Model of In-Context Learning Using Linear Attention, Deeper Learning Blog, by Mary Letey

Part I: Parameterizations for Predictable Scaling

Lazy vs Rich Limits

As discussed in Andrea's lectures, previous work identified different behaviors arising from infinite (width, depth, attention head, ….) limits of networks with different parameterizations and initializations

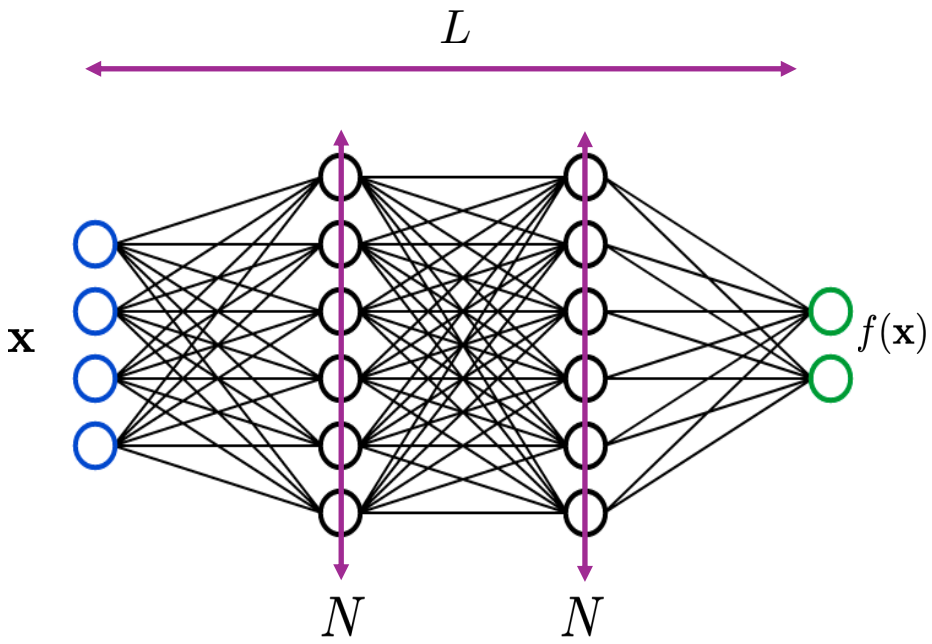
- lazy/static (NTK – Kernel limit): Network does not adapt its internal representations to data. Worse performance compared to rich regime in practice. Easier to analyze (see Andrea's talk), but not realistic.

  (Chizat and Bach, 2018; Jacot, Gabriel, Hongler, 2018)

- feature learning/rich (mean field limit): Network learns internal representations. Better performance and more realistic, but harder to analyze. There may be many rich limits for a given architecture.

  (Two-layer networks: Rotskoff, Vanden-Eijnden 2018; Mei, Montanari, Nguyen, 2018)
  (Deep MLPs: Yang & Hu 2020; Bordelon and Pehlevan, 2022)
  (ResNets: Bordelon et al., 2023; Yang et al., 2023)
  (Transformers: Bordelon et al., 2025; Dey et al. 2025)

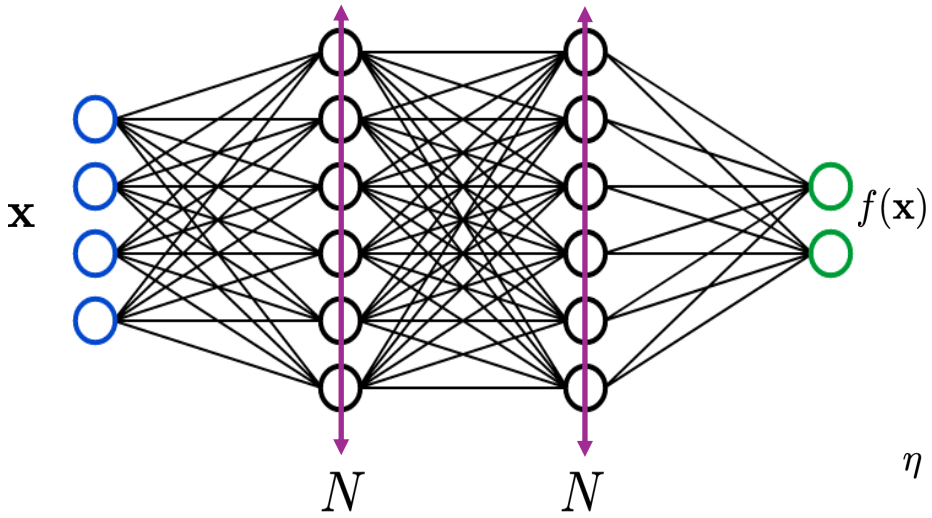
Case 1: Infinite-width limits for MLPs



Let's first fix depth (L) at a finite number, and take width (N) to infinity

Different parameterizations for for MLPs

$W_{ij} \sim \mathcal{N}(0,1)$ $\quad \frac{W_{ij}}{\sqrt{N}}$ $\quad \frac{W_{ij}}{\gamma_0 N}$



We will derive these scalings in a few slides.

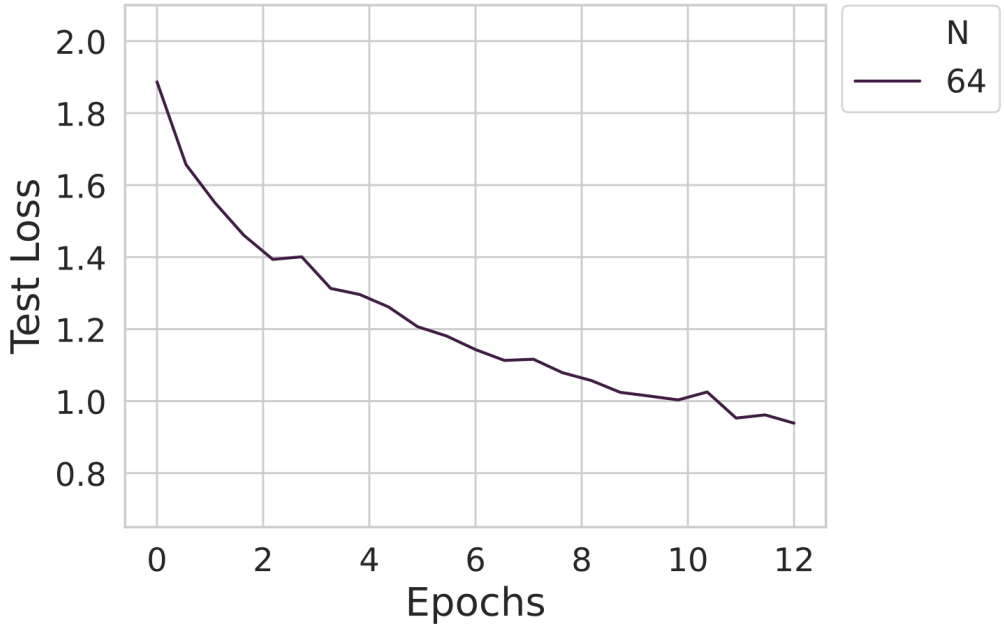
$\eta = \eta_0 \gamma_0^2 N$

γ_0 controls speed of representation learning (Chizat and Bach, 2019; Geiger et al, 2020)

Neural Tangent parameterization (Jacot et al, 2018; Lee et al, 2019): $\gamma_0 = \Theta_N(1/\sqrt{N})$

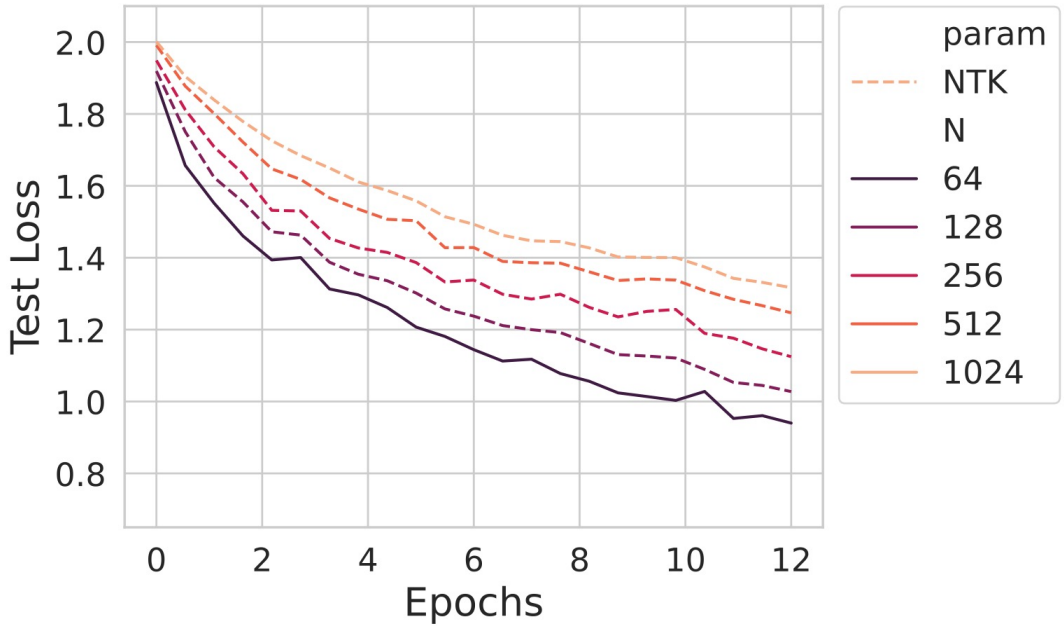
Mean field / Maximal Update (mu) parameterization (Mei et al 2018, Yang et al 2021): $\gamma_0 = \Theta_N(1)$

Example: Mean-field/muP vs NTK parameterizations



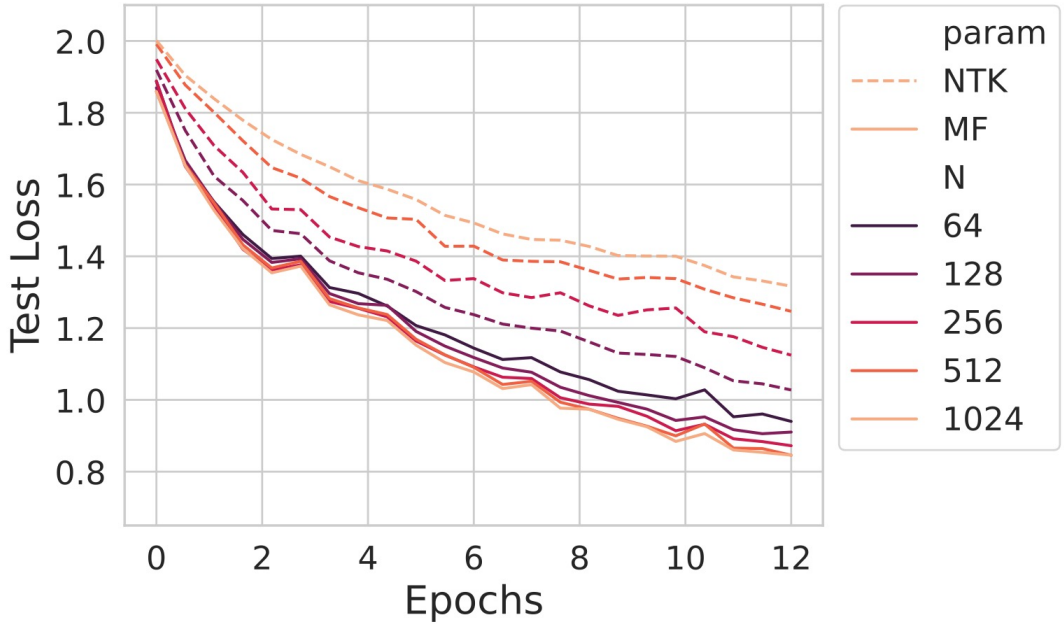
Depth 12 ResNet on CIFAR-10 SGD training

Example: Mean-field/muP vs NTK parameterizations



Depth 12 ResNet on CIFAR-10 SGD training

Example: Mean-field/muP vs NTK parameterizations



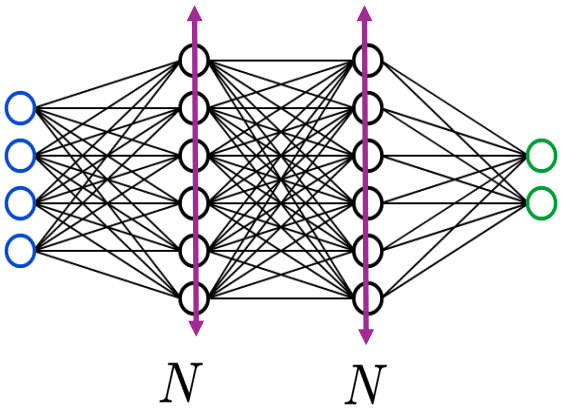
Depth 12 ResNet on CIFAR-10 SGD training

Parameterizations

Loss/Data: $\mathcal{L}(\boldsymbol{\theta}; \mathcal{D}), \quad \mathcal{D} = \{(\mathbf{x}_\mu, y_\mu)\}_{\mu=1}^{P}, \quad \mathbf{x} \in \mathbb{R}^D$

Training: $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla \mathcal{L}(\boldsymbol{\theta}_t)$

Possible parameterizations and initializations:



$$f_\mu = \frac{1}{\gamma} h_\mu^{(L)},$$

$$h_\mu^{(L)} = \frac{1}{N^{a_L}} \mathbf{w}^{(L)} \cdot \phi(\mathbf{h}_\mu^{(L-1)}) \ , \qquad w_i^{(L)} \sim \mathcal{N}\left(0, \frac{1}{N^{b_L}}\right),$$

$$\mathbf{h}_\mu^{(\ell)} = \frac{1}{N^{a_\ell}} \mathbf{W}^{(\ell)} \phi(\mathbf{h}_\mu^{(\ell-1)}) \ , \qquad \mathbf{W}_{ij}^{(\ell)} \sim \mathcal{N}\left(0, \frac{1}{N^{b_\ell}}\right),$$

$$\mathbf{h}_\mu^{(1)} = \frac{1}{N^{a_1}} \frac{1}{D^{1/2}} \mathbf{W}^{(1)} \mathbf{x}_\mu \ , \qquad W_{ij}^{(1)} \sim \mathcal{N}\left(0, \frac{1}{N^{b_1}}\right),$$

$$\eta = \eta_0 \gamma^2 N^{-c},$$

$$\gamma = \gamma_0 N^d$$

How should we choose {a}, {b}, c, d?

Desiderata for feature learning infinite limits

As we scale width (N), we demand the following:

1. **Stable Initialization:** Hidden layers and output remain stable at initialization

$$h_i^{(\ell)} = \Theta_N(1), \quad f = O_N(1)$$

2. **Training:** Outputs evolve in finite time

$$\Delta_t f(\mathbf{x}) \equiv f(\mathbf{x}; \boldsymbol{\theta}_t) - f(\mathbf{x}; \boldsymbol{\theta}_{t-1}) = \Theta_N(1)$$

3. **Feature Learning:** Features evolve in finite time (this doesn't happen in NTK parameterization)

$$\Delta_t h_i^{(\ell)}(\mathbf{x}) \equiv h_i^{(\ell)}(\mathbf{x}; \boldsymbol{\theta}_t) - h_i^{(\ell)}(\mathbf{x}; \boldsymbol{\theta}_{t-1}) = \Theta_N(1)$$

Yang & Hu 2020; Bordelon and Pehlevan, 2022

Stable Initialization

$$\left\langle h_{\mu,i}^{(1)} \right\rangle = \frac{1}{N^{a_1}} \frac{1}{\sqrt{D}} \sum_k \left\langle W_{ik}^{(1)}(0) \right\rangle x_{\mu,k} = 0. \qquad \langle q(\boldsymbol{\theta}) \rangle \equiv \mathbb{E}_{\boldsymbol{\theta}} \left[q(\boldsymbol{\theta}) \right]$$

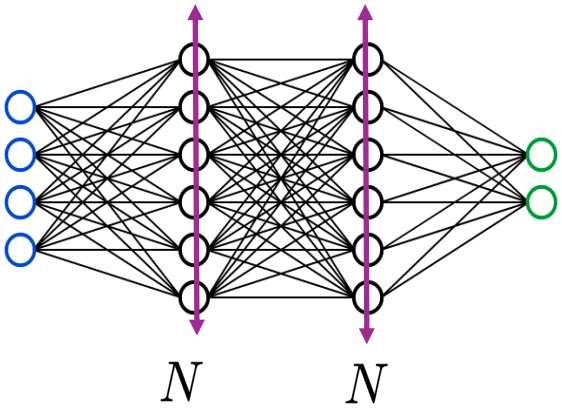
$$\left\langle h_{\mu,i}^{(1)} h_{\nu,j}^{(1)} \right\rangle = \frac{1}{N^{2a_1}} \frac{1}{D} \sum_{kk'} \left\langle W_{ik}^{(1)}(0) W_{jk'}^{(1)}(0) \right\rangle x_{\mu,k} x_{\nu,k'} = \delta_{ij} \frac{1}{N^{2a_1+b_1}} \frac{1}{D} \sum_{k=1}^{D} x_{\mu,k} x_{\nu,k} \qquad \Longrightarrow \quad \boxed{2a_1 + b_1 = 0}$$

$$\left\langle h_{\mu,i}^{(2)} \right\rangle = \frac{1}{N^{a_2}} \sum_k \left\langle W_{ik}^{(2)}(0) \phi(h_{\mu,k}^{(1)}) \right\rangle = \frac{1}{N^{a_2}} \sum_k \left\langle W_{ik}^{(2)}(0) \right\rangle \left\langle \phi(h_{\mu,k}^{(1)}) \right\rangle = 0$$

$$\begin{aligned} \left\langle h_{\mu,i}^{(2)} h_{\nu,j}^{(2)} \right\rangle &= \frac{1}{N^{2a_2}} \sum_{k,k'} \left\langle W_{ik}^{(2)}(0) W_{jk'}^{(2)}(0) \phi(h_{\mu,k}^{(1)}) \phi(h_{\nu,k'}^{(1)}) \right\rangle \\ &= \frac{1}{N^{2a_2}} \sum_{k,k'} \left\langle W_{ik}^{(2)}(0) W_{jk'}^{(2)}(0) \right\rangle \left\langle \phi(h_{\mu,k}^{(1)}) \phi(h_{\nu,k'}^{(1)}) \right\rangle \\ &= \delta_{ij} \frac{1}{N^{2a_2+b_2-1}} \frac{1}{N} \sum_{k=1}^{N} \left\langle \phi(h_{\mu,k}^{(1)}) \phi(h_{\nu,k}^{(1)}) \right\rangle = \delta_{ij} \frac{1}{N^{2a_2+b_2-1}} \left\langle \phi(h_{\mu,1}^{(1)}) \phi(h_{\nu,1}^{(1)}) \right\rangle . \end{aligned}$$

$$\Longrightarrow \quad \boxed{2a_\ell + b_\ell = 1, \quad l = 2, \ldots, L.}$$

muP/ mean field parameterization



$$f_\mu = \frac{1}{\gamma} h_\mu^{(L)},$$

$$h_\mu^{(L)} = \frac{1}{N^{a_L}} \mathbf{w}^{(L)} \cdot \phi(\mathbf{h}_\mu^{(L-1)}) \ , \qquad w_i^{(L)} \sim \mathcal{N}\left(0, \frac{1}{N^{b_L}}\right),$$

$$\mathbf{h}_\mu^{(\ell)} = \frac{1}{N^{a_\ell}} \mathbf{W}^{(\ell)} \phi(\mathbf{h}_\mu^{(\ell-1)}) \ , \qquad \mathbf{W}_{ij}^{(\ell)} \sim \mathcal{N}\left(0, \frac{1}{N^{b_\ell}}\right),$$

$$\mathbf{h}_\mu^{(1)} = \frac{1}{N^{a_1}} \frac{1}{D^{1/2}} \mathbf{W}^{(1)} \mathbf{x}_\mu \ , \qquad W_{ij}^{(1)} \sim \mathcal{N}\left(0, \frac{1}{N^{b_1}}\right),$$

$$\eta = \eta_0 \gamma^2 N^{-c},$$

$$\gamma = \gamma_0 N^d$$

Demand the following:

1. Stable initialization
2. Training
3. Feature Learning

$$\implies 2a_\ell + b_\ell = 1 \quad \text{for } \ell \in \{2, ..., \mathrm{L}\},\ 2\mathrm{a}_1 + \mathrm{b}_1 = 0$$

$$\implies 2a_\ell + c = 1 \quad \text{for } \ell \in \{2, ..., \mathrm{L}\},\ 2\mathrm{a}_1 + \mathrm{c} = 0$$

$$\implies d = \frac{1}{2}$$

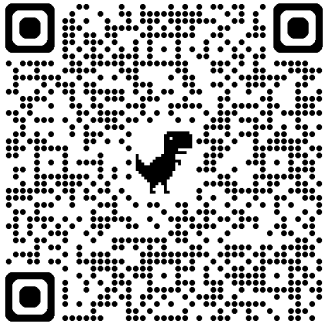
Yang & Hu 2020; Bordelon and Pehlevan, 2022

Full details of these calculations can be found in

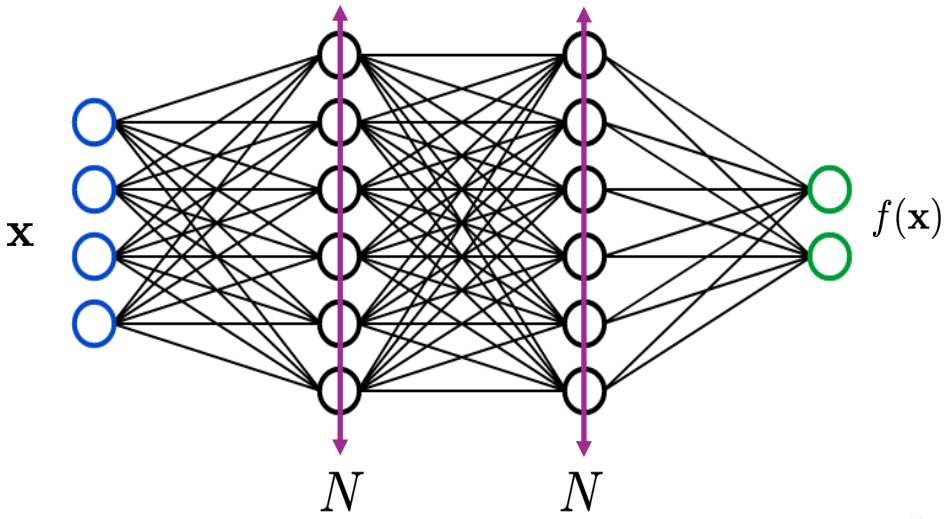
Lecture Notes on Infinite-Width Limits of Neural Networks

Cengiz Pehlevan and Blake Bordelon

June 2023



$W_{ij} \sim \mathcal{N}(0,1)$ $\quad \frac{W_{ij}}{\sqrt{N}} \quad \frac{W_{ij}}{\gamma_0 N}$



$\eta = \eta_0 \gamma_0^2 N$

γ_0 controls speed of representation learning (Chizat and Bach, 2019; Geiger et al, 2020)

Neural Tangent (Jacot et al, 2018; Lee et al, 2019): $\gamma_0 \sim \mathcal{O}(1/\sqrt{N})$

Mean field scaling (Mei et al 2018, Yang et al 2021): $\gamma_0 \sim \mathcal{O}(1)$

Hyperparameter transfer

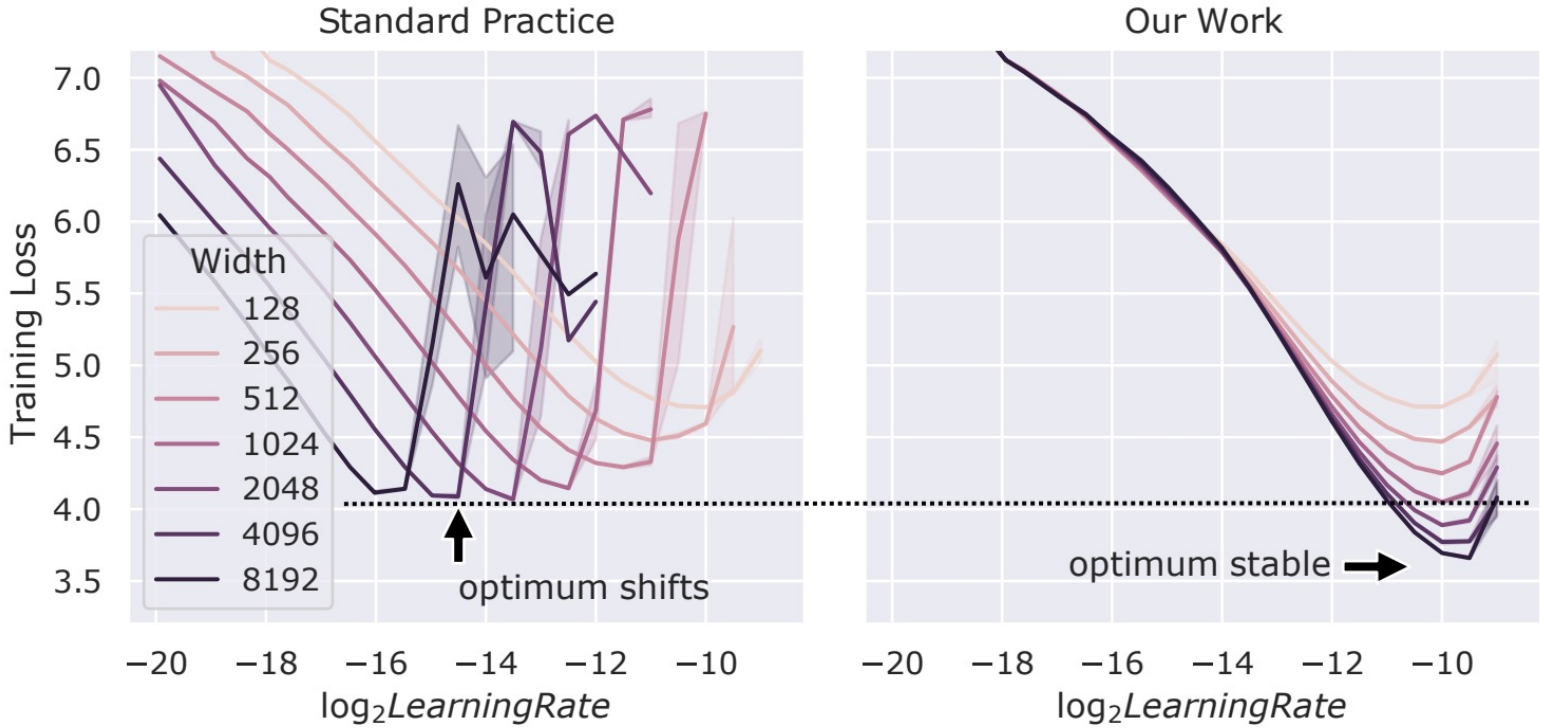
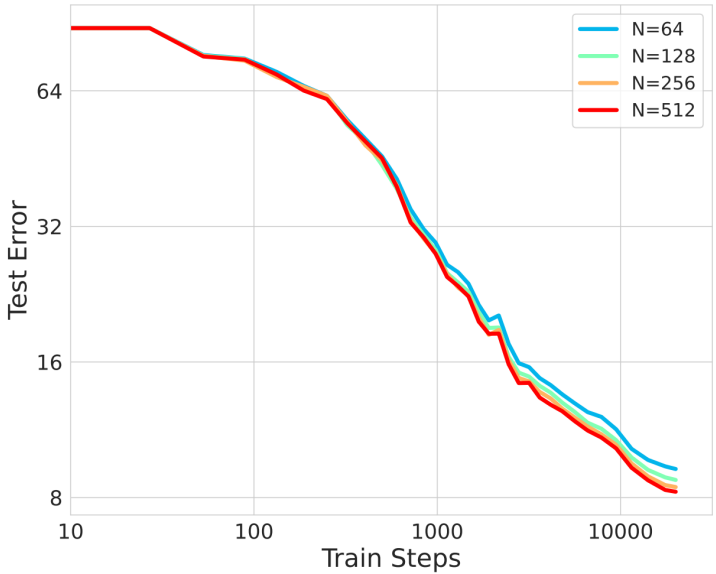
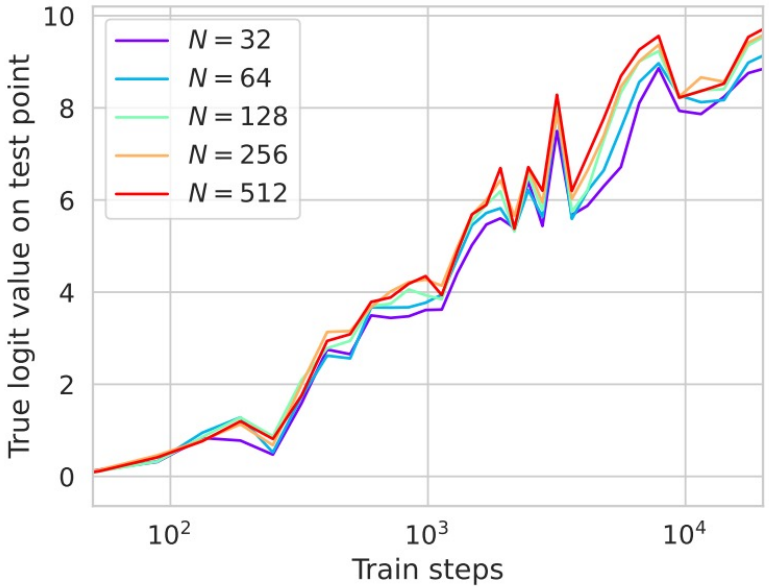


Figure from Yang et al., 2022

Why does hyperparameter transfer work? Consistent behavior across scales

ResNet18 on CIFAR-5M, online training, batch size 250

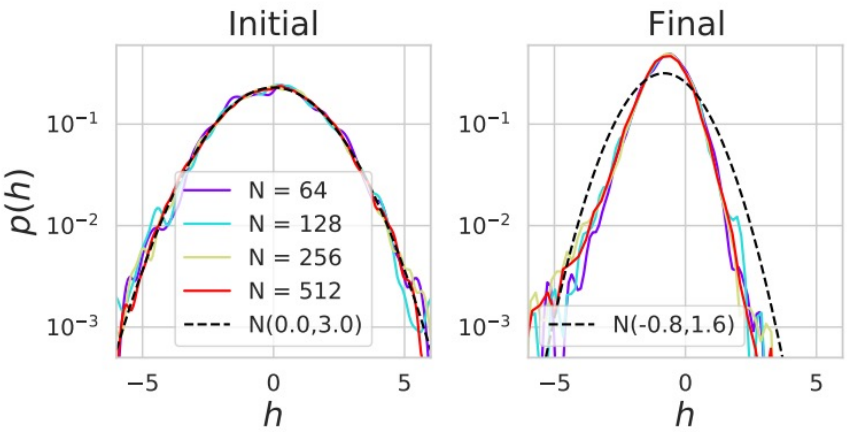




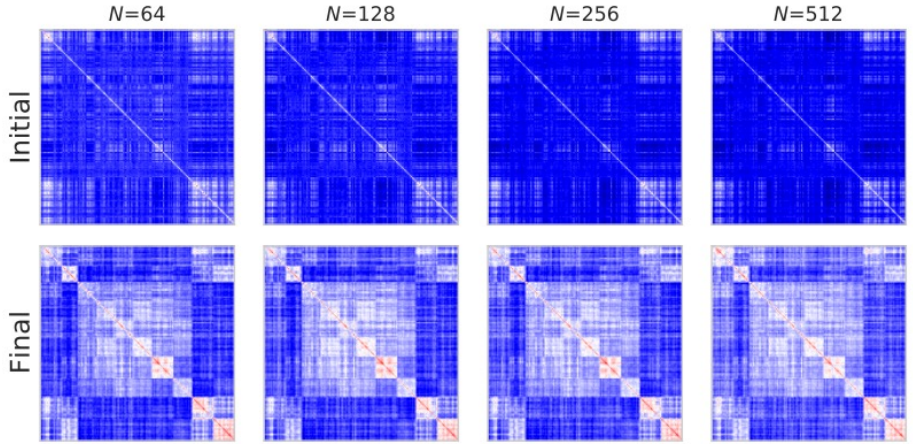
More experiments in Vyas et al, NeurIPS, 2023

Consistent behavior across widths

ResNet18 on CIFAR-5M, online training, batch size 250



Pre-activation histograms

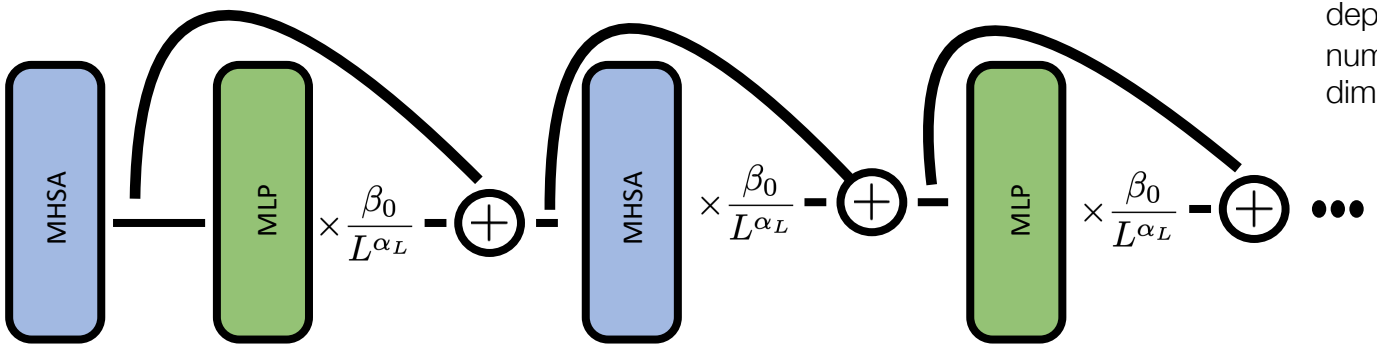


Last layer similarity matrices

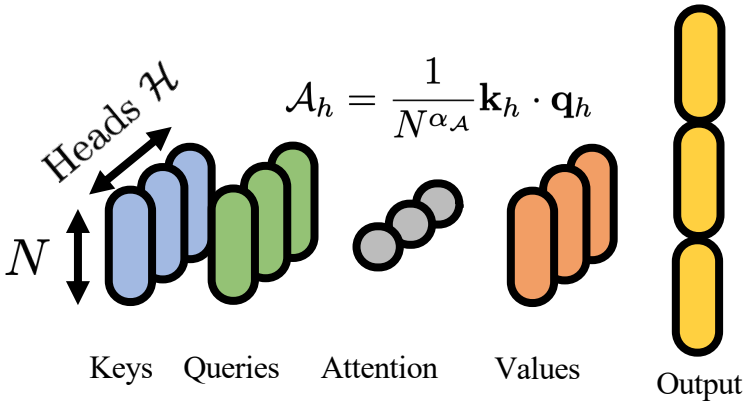
More experiments in Vyas et al, NeurIPS, 2023

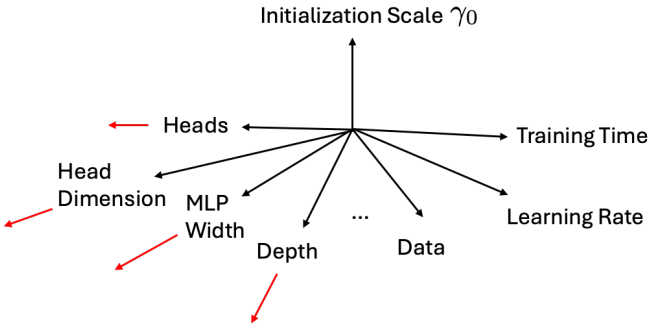
Case 2: Transformers Parameterizations for Predictable Scaling

Many choices:
depth (L),
number of heads (H),
dimension per head (N)



Multihead Self-Attention (MHSA) Layer





Theory

DEPTHWISE HYPERPARAMETER TRANSFER IN RESIDUAL NETWORKS: DYNAMICS AND SCALING LIMIT

Blake Bordelon[*] [¶] **, Lorenzo Noci**[*] [♯] **, Mufan (Bill) Li** [§] **, Boris Hanin**[†] [§] **& Cengiz Pehlevan**[†] [¶]
[¶] Harvard University
[♯] ETH Zürich
[§] Princeton University

Infinite Limits of Multi-head Transformer Dynamics

Blake Bordelon, Hamza Chaudhry, Cengiz Pehlevan
John A. Paulson School of Engineering and Applied Sciences
Center for Brain Science
Kempner Institute for the Study of Natural and Artificial Intelligence
Harvard University
Cambridge, MA 02138
blake_bordelon@g.harvard.edu
hchaudhry@g.harvard.edu
cpehlevan@seas.harvard.edu

Feature Learning in Infinite-Depth Neural Networks

Greg Yang[*] xAI
Dingli Yu[*] Princeton Language and Intelligence
Chen Zhu Nvidia
Soufiane Hayou[†] Simons Institute UC Berkeley

Don't be lazy: CompleteP enables compute-efficient deep transformers

Nolan Dey[*] Cerebras Systems
Bin Claire Zhang[*] Cerebras Systems
Lorenzo Noci ETH Zurich Princeton University
Mufan Li Princeton University
Blake Bordelon Harvard University
Shane Bergsma Cerebras Systems
Cengiz Pehlevan Harvard University Kempner Institute
Boris Hanin Princeton University
Joel Hestness Cerebras Systems

Extension, refined desiderata, and experimental verification

Setup

$$\mathbf{h}^{\ell+1} = \mathbf{h}^{\ell} + L^{-\alpha}\ \mathcal{F}_{\ell}(\mathbf{h}^{\ell}; \boldsymbol{\theta}^{\ell}), \qquad \ell = \{1, \ldots, L\}$$

Fixed depth network such as an
Attention or MLP block

$$\boldsymbol{\theta}^{\ell} \ \leftarrow \ \boldsymbol{\theta}^{\ell} + \Delta\boldsymbol{\theta}^{\ell}, \qquad \Delta\boldsymbol{\theta}^{\ell} = -\eta^{\ell}\ \cdot\ \mathbf{g}^{\ell}_{\mathrm{AdamW}}$$

Desideratum 1 (Stable Initialization)**.** *Hidden layers and output remain stable at initialization. More precisely, for all layers* $\ell \in [L]$, $\frac{1}{N}\|\mathbf{h}^\ell\|_2^2 \in \Theta(1)$ *and* $f \in O(1)$, *as* $N \to \infty, L \to \infty$.

$$\boxed{\implies \alpha > 1/2}$$

Quick derivation: $\mathbf{h}^{\ell+1} = \mathbf{h}^\ell + \frac{1}{L^\alpha}\mathbf{W}^\ell \phi(\mathbf{h}^\ell), \qquad W_{ij}^\ell \sim \mathcal{N}(0, \frac{\sigma_W^2}{N})$

At large N $\qquad H^{\ell+1} = H^\ell + L^{-2\alpha}\sigma_W^2 \mathbb{E}_{h\sim\mathcal{N}(0,H^\ell)}\phi(h)^2, \qquad H^\ell \equiv \frac{1}{N}|\mathbf{h}^\ell|^2$

Example: $\phi(h) = \mathrm{ReLU}(h)$ $\qquad H^{\ell+1} = H^\ell + \frac{\sigma_W^2}{2}L^{-2\alpha}H^\ell = \left[1 + \frac{\sigma_W^2}{2}L^{-2\alpha}\right]^\ell H^0$

$$\lim_{L\to\infty} H^L = \begin{cases} H^0 & \alpha > \frac{1}{2} \\ \exp\left(\frac{\sigma_W^2}{2}\right) H^0 & \alpha = \frac{1}{2} \\ \infty & \alpha < \frac{1}{2} \end{cases}.$$

Desideratum 2 (Maximal Residual Stream Update)**.** *Each residual block's weights should contribute order* $1/L$ *to feature movements, and each non-residual block should contribute constant order. More precisely, for all* $\ell \in [L-1]$*, each block's parameter update* $\boldsymbol{\theta}^\ell \mapsto \boldsymbol{\theta}^\ell + \Delta\boldsymbol{\theta}^\ell$ *should contribute the change* $\frac{1}{N}\|\Delta_{\boldsymbol{\theta}^\ell}\mathbf{h}^{\ell+1}\|_2^2 \in \Theta(1/L)$*. Moreover, for the embedding and unembedding layers we require* $\frac{1}{N}\|\Delta\mathbf{W}^0 x\|_2^2 \in \Theta(1)$ *and* $\frac{1}{N}\|\Delta\mathbf{W}^L\mathbf{h}^L\|_2^2 \in \Theta(1)$*.*

$$\boxed{\implies \eta = \Theta(L^{1-\alpha})}$$

Desideratum 3 (Complete Feature Learning). *The network parameterization satisfies complete feature learning, i.e. neither the hidden layers $\{h^\ell\}_{\ell\in[L]}$ nor the model output f are lazy with respect to any subset of model parameters.*

$$h^{\text{lin},\theta}(\boldsymbol{\theta},\boldsymbol{\theta}_0) = \mathbf{h}(\boldsymbol{\theta}_0) + \nabla_{\boldsymbol{\theta}}\mathbf{h}(\boldsymbol{\theta})|_{\boldsymbol{\theta}_0}\cdot(\boldsymbol{\theta}-\boldsymbol{\theta}_0)$$

Definition. *We say a layer $\mathbf{h}^\ell$ is **lazy** with respect to a subset of parameters $\theta \subset \{\theta_j\}_{j<\ell}$ if at finite depth and width, $\mathbf{h}^\ell$ is not linear in θ and the change $\Delta_\theta\mathbf{h}^\ell$ at initialization from updating only θ (i.e. replacing $\theta \mapsto \theta + \Delta\theta$) is asymptotically the same as the change to the linearization of h:*

$$\frac{|\Delta_\theta\mathbf{h}^\ell - \Delta_\theta\mathbf{h}_\ell^{lin,\theta}|}{|\Delta_\theta\mathbf{h}_\ell^{lin,\theta}|} = o(1)\,, \qquad \textit{as } N, L \to \infty\,.$$

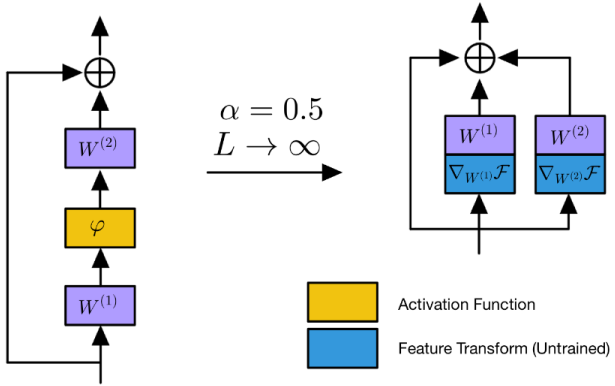
$$\boxed{\implies \alpha = 1}$$

Simple Block Depth 2 Example.

Consider $N = 1$ $\quad h^{\ell+1} = h^\ell + L^{-\alpha}\, W^\ell_{(2)} W^\ell_{(1)} h^\ell, \qquad \ell = \{1, \dots, L\} \qquad (W^\ell_{(1)}, W^\ell_{(2)}) = \boldsymbol{\theta}^\ell \mapsto \boldsymbol{\theta}^\ell + \Delta\boldsymbol{\theta}^\ell$

Taylor expand: $\quad \Delta_{\boldsymbol{\theta}^\ell} h^{\ell+1} = \nabla_{\boldsymbol{\theta}^\ell} h^{\ell+1} \cdot \Delta\boldsymbol{\theta}^\ell + \frac{1}{2} \Delta\boldsymbol{\theta}^{\ell\top} \nabla^2_{\boldsymbol{\theta}^\ell} h^{\ell+1} \Delta\boldsymbol{\theta}^\ell$

$$= \underbrace{L^{-\alpha}(W^\ell_{(2)} h^\ell \underbrace{\Delta W^\ell_{(1)}}_{L^{\alpha-1}} + W^\ell_{(1)} h^\ell \underbrace{\Delta W^\ell_{(2)}}_{L^{\alpha-1}})}_{L^{-1}} + \underbrace{L^{-\alpha} h^\ell \underbrace{\Delta W^\ell_{(1)} \Delta W^\ell_{(2)}}_{L^{2(\alpha-1)}}}_{L^{\alpha-2}} \cdot \implies \alpha = 1$$



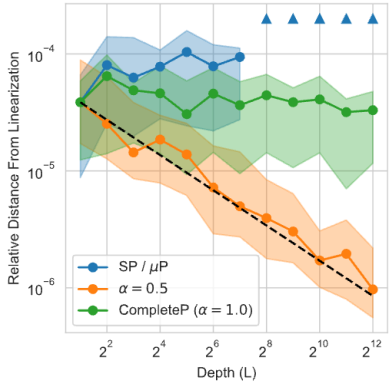


Table 1: Summary of SP, μP, and $\alpha \in \{0.5, 1\}$ for a pre-LN transformer language model. Terms related to width and depth control are highlighted in orange and green respectively. Additional tunable parameters are highlighted in blue. *Hidden* refers to all linear layers in the transformer backbone.

Parameterization	SP	μP	$\alpha \in \{0.5, 1\}$
Emb. Init. Var. Emb. LR (AdamW)	σ^2_{base} η_{base}	σ^2_{base} η_{base}	σ^2_{base} η_{base}
Pre-LN Init. Var. Pre-LN LR (AdamW)	σ^2_{base} η_{base}	σ^2_{base} η_{base}	σ^2_{base} $\eta_{\text{base}} {m_L}^{\alpha-1}$
Hidden Init. Var. Hidden LR (AdamW) Hidden Bias LR (AdamW) Hidden WD (AdamW)	σ^2_{base} η_{base} η_{base} λ_{base}	$\sigma^2_{\text{base}} \cdot m_N^{-1}$ $\eta_{\text{base}} \cdot m_N^{-1}$ η_{base} $\lambda_{\text{base}} \cdot m_N$	$\sigma^2_{\text{base}} \cdot m_N^{-1}$ $\eta_{\text{base}} \cdot m_N^{-1} \cdot {m_L}^{\alpha-1}$ $\eta_{\text{base}} {m_L}^{\alpha-1}$ $\lambda_{\text{base}} \cdot m_N$
MHA Residual MLP Residual	$\mathbf{X}^l + \text{MHA}(\text{LN}(\mathbf{X}^l))$ $\mathbf{Z}^l + \text{MLP}(\text{LN}(\mathbf{Z}^l))$	$\mathbf{X}^l + \text{MHA}(\text{LN}(\mathbf{X}^l))$ $\mathbf{Z}^l + \text{MLP}(\text{LN}(\mathbf{Z}^l))$	$\mathbf{X}^l + {m_L}^{-\alpha} \cdot \text{MHA}(\text{LN}(\mathbf{X}^l))$ $\mathbf{Z}^l + {m_L}^{-\alpha} \cdot \text{MLP}(\text{LN}(\mathbf{Z}^l))$
Final-LN Init. Var. Final-LN LR (AdamW)	σ^2_{base} η_{base}	σ^2_{base} η_{base}	σ^2_{base} η_{base}
Unemb. Init. Var. Unemb. LR (AdamW) Unemb. Fwd.	σ^2_{base} η_{base} $\mathbf{X}^L \mathbf{W}^\top_{\text{unemb}}$	σ^2_{base} η_{base} $\mathbf{X}^L \mathbf{W}^\top_{\text{unemb}} \cdot m_N^{-1}$	σ^2_{base} η_{base} $\mathbf{X}^L \mathbf{W}^\top_{\text{unemb}} \cdot m_N^{-1}$
AdamW ϵ (Residual blocks) AdamW ϵ (Emb. & Unemb.)	ϵ_{base} ϵ_{base}	$\epsilon_{\text{base}} \cdot m_N^{-1}$ ϵ_{base}	$\epsilon_{\text{base}} \cdot m_N^{-1} \cdot {m_L}^{-\alpha}$ ϵ_{base}

Yang & Hu 2020

Bordelon et al., 2023; Yang et al., 2023; Bordelon et al., 2025; Dey et al. 2025

Practical benefits

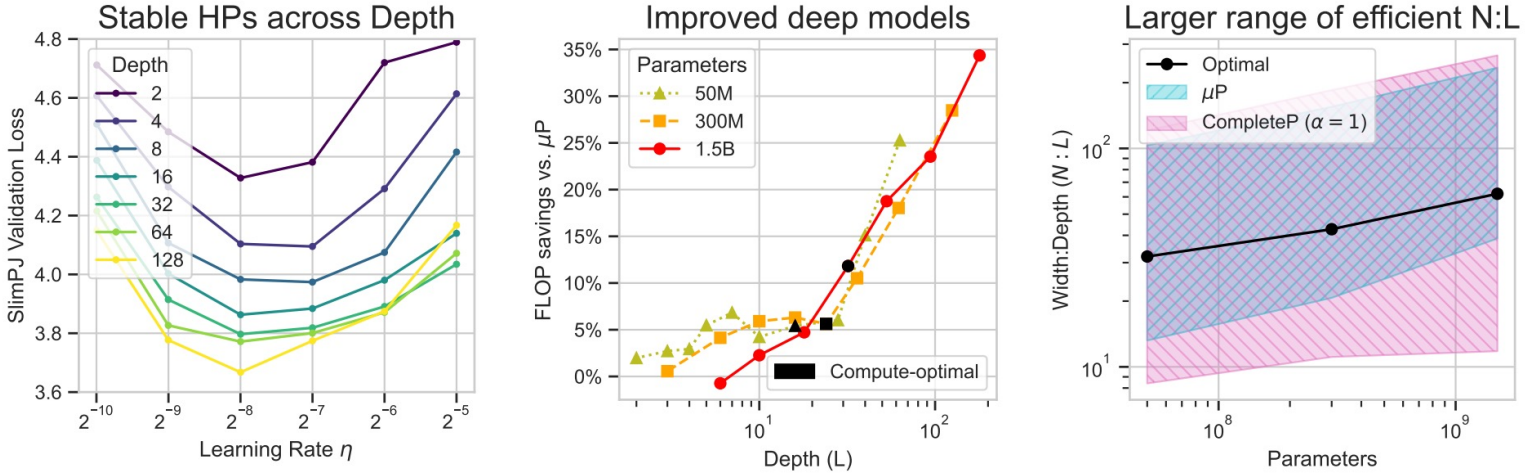


Figure 1: We introduce **CompleteP**, which offers depth-wise HP transfer (**Left**), FLOP savings when training deep models (**Middle**), and a larger range of compute-efficient width/depth ratios (**Right**).

Conclusion for Part 1

- Different parameterizations and initializations lead to different behaviors in scaling limits
- Parameterizations that allow predictable and consistent scaling offer hyperparameter transfer benefits
- More work to be done as new architectural changes come in (e.g. mixture of experts)

Resources

LECTURE NOTES, REVIEWS AND BLOG POSTS

Replica Method for the Machine Learning Theorist - Part 1, by Blake Bordelon, Haozhe Shan, Abdul Canatar, Boaz Barak, Cengiz Pehlevan

Replica Method for the Machine Learning Theorist - Part 2, by Blake Bordelon, Haozhe Shan, Abdul Canatar, Boaz Barak, Cengiz Pehlevan

Lecture notes on the replica method for Wishart matrix eigenvalues, by Jacob Zavatone-Veth

A brief introduction to the neural network Gaussian process from the perspective of mean field theory, by Jacob Zavatone-Veth

Lecture Notes on Infinite-Width Limits of Neural Networks, Cengiz Pehlevan and Blake Bordelon, prepared for 2023 Princeton ML Theory Summer School

Infinite Limits of Neural Networks, Deeper Learning Blog, by Alex Atanasov, Blake Bordelon and Cengiz Pehlevan

A Dynamical Model of Neural Scaling Laws, Deeper Learning Blog, by Blake Bordelon, Alex Atanasov and Cengiz Pehlevan

Scaling and renormalization in high-dimensional regression, by Alex Atanasov, Jacob Zavatone-Veth, Cengiz Pehlevan

Solvable Model of In-Context Learning Using Linear Attention, Deeper Learning Blog, by Mary Letey

Part II - Dynamical Mean Field Theory Description of Feature Learning Dynamics

See Andrea Montanari's Turing Lecture 3
for results on two-layer MLPs.
I will discuss deeper MLPs (and point to
references for other architectures).

Dynamical Mean Field Theory

Dynamical mean-field theory (DMFT) offers a powerful theoretical approach for reducing the equations of motion of high-dimensional system into a single equation governing an effective, or “mean-field,” particle.

Why is this useful?

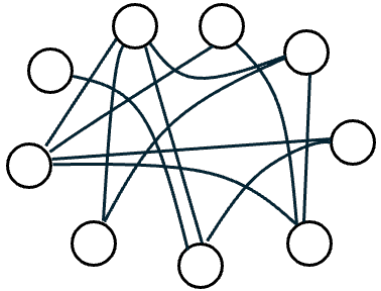
1. Mean field equations can be used to gain various analytical insights into the problem.
2. Mean field equations can provide computational gains compared to simulating the full high dimensional system

Specifically for this context:

3. Scaling the size of neural networks give better performing models. Hence, DMFT describes the “best” model for a given architecture.

Primer on Dynamical Mean Field Theory

Random Coupled Systems in High Dimensions



Spin Glass Example: $\mathcal{H}(\{s_i\}) = -\frac{1}{2\sqrt{N}} \sum_{ij} J_{ij} s_i s_j \qquad J_{ij} = J_{ji} \sim \mathcal{N}(0,1)$

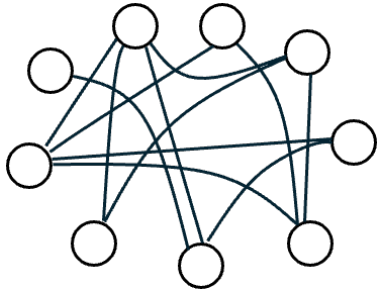
Langevin Dynamics on sphere $\partial_t s_i(t) = \frac{1}{\sqrt{N}} \sum_{j=1}^{N} J_{ij} s_j(t) - \lambda s_i(t) + \eta_i(t)$

Gross, Mezard, Kirkpatrick, Thirumalai, Crisanti, Horner, Sommers, Sompolinsky,….

Primer on Dynamical Mean Field Theory

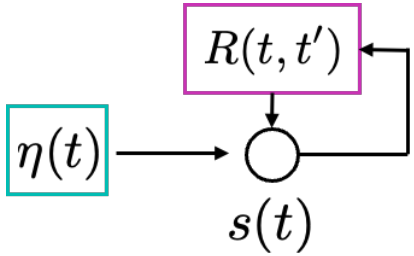
Sompolinsky & Zippelius ’82, Kurchan & Cugliandolo ’93, Crisanti, Horner & Summers ‘93, Bouchaud et al ‘97

Random Coupled -> Uncoupled System in the Limit



$$\sim$$

$$N \to \infty$$



$$\partial_t s(t) = -\lambda(t)s(t) + \underbrace{\eta(t)}_{\text{colored noise}} + \underbrace{\int dt' R(t,t')s(t')}_{\text{memory term}}$$

Correlation and Response Form Closed System from Single Site Picture

$$\eta(t) \sim \mathcal{GP}(0, C(t,t')) \qquad C(t,t') = \langle s(t)s(t') \rangle \quad R(t,t') = \left\langle \frac{\delta s(t)}{\delta \eta(t')} \right\rangle$$

Many theoretical methods give this result

1. Saddle point of a Martin Siggia Rose Path integral $$Z = \int dC dR \exp\left(-N\mathcal{S}(C,R)\right)$$

2. Cavity (add new site) argument, compute self-feedback through other sites

A great first reading on DMFT

Building Intuition for Dynamical Mean-Field Theory: A Simple Model and the Cavity Method

A tutorial oriented to the biophysics community

Emmy Blumenthal

Princeton University Department of Physics, Princeton, NJ 08540, USA

July 23, 2025

Can we apply this idea to neural networks?

Loss/Data: $\mathcal{L}(\boldsymbol{\theta}; \mathcal{D}), \quad \mathcal{D} = \{(\mathbf{x}_\mu, y_\mu)\}_{\mu=1}^{P}, \quad \mathbf{x} \in \mathbb{R}^D$

Training: $$\frac{d\boldsymbol{\theta}}{dt} = -\frac{\eta}{P} \frac{\partial \mathcal{L}(\boldsymbol{\theta}; \mathcal{D})}{\partial \boldsymbol{\theta}}$$

Is there a mean field description of this dynamics?

Dynamical mean-field theory of learning dynamics of *feature learning deep* networks in the infinite-width (and depth) limit

Self-Consistent Dynamical Field Theory of Kernel Evolution in Wide Neural Networks

Blake Bordelon & Cengiz Pehlevan

NeurIPS 2022

THE INFLUENCE OF LEARNING RULE ON REPRESENTATION DYNAMICS IN WIDE NEURAL NETWORKS

Blake Bordelon & Cengiz Pehlevan

ICLR, 2023

DEPTHWISE HYPERPARAMETER TRANSFER IN RESIDUAL NETWORKS: DYNAMICS AND SCALING LIMIT

Blake Bordelon* ¶ , Lorenzo Noci*♯ , Mufan (Bill) Li § , Boris Hanin† § & Cengiz Pehlevan† ¶
¶ Harvard University
♯ ETH Zürich
§ Princeton University

ICLR 2024

Infinite Limits of Multi-head Transformer Dynamics

Blake Bordelon, Hamza Chaudhry, Cengiz Pehlevan
John A. Paulson School of Engineering and Applied Sciences
Center for Brain Science
Kempner Institute for the Study of Natural and Artificial Intelligence
Harvard University
Cambridge, MA 02138
blake_bordelon@g.harvard.edu
hchaudhry@g.harvard.edu
cpehlevan@seas.harvard.edu

NeurIPS 2024

Similar limits considered in two-layer networks by Rotskoff, Vanden-Eijnden 2018; Mei, Montanari, Nguyen, 2018; and using the “Tensor Programs” Yang & Hu 2020

Gradient-flow in feature space

Gradient-flow can be rewritten in “feature/activation” space *without reference to weights except at initialization*. Key quantities are layer-wise feature and gradient kernels.

$$\Phi^{(\ell)}_{\mu\nu}(t,s) = \frac{1}{N}\phi(\mathbf{h}^{(\ell)}_{\mu}(t)) \cdot \phi(\mathbf{h}^{(\ell)}_{\nu}(s)) \ , \qquad G^{(\ell)}_{\mu\nu}(t,s) = \frac{1}{N}\mathbf{g}^{(\ell)}_{\mu}(t) \cdot \mathbf{g}^{(\ell)}_{\nu}(s) \qquad \mathbf{g}^{(\ell)}_{\mu} \equiv \sqrt{N}\frac{\partial h^{(L)}_{\mu}}{\partial \mathbf{h}^{(\ell)}_{\mu}}$$

$$\frac{\partial f_{\mu}}{\partial t} = \frac{\eta_0}{P}\sum_{\nu}\Delta_{\nu}(t)\left[\Phi^{(L-1)}_{\mu\nu} + \sum_{\ell=1}^{L-1} G^{(\ell)}_{\mu\nu}\Phi^{(\ell-1)}_{\mu\nu}\right] \qquad \Delta_{\mu} = -\frac{\partial \mathcal{L}}{\partial f_{\mu}}$$

$N \to \infty$ limit

1. Feature and Gradient kernels concentrate but evolve in time.

$$\Phi^{(\ell)}_{\mu\nu}(t,s) = \frac{1}{N}\phi(\mathbf{h}^{(\ell)}_{\mu}(t)) \cdot \phi(\mathbf{h}^{(\ell)}_{\nu}(s)) \ , \qquad G^{(\ell)}_{\mu\nu}(t,s) = \frac{1}{N}\mathbf{g}^{(\ell)}_{\mu}(t) \cdot \mathbf{g}^{(\ell)}_{\nu}(s)$$

2. Distribution of fields factorize over sites and layers.

$$p(\{h^{\ell}_i, g^{\ell}_i\}) = \prod_{\ell=1}^{L}\prod_{i=1}^{N} p_{\ell}(h^{\ell}_i, g^{\ell}_i)$$

3. Population averages are replaced by single-site averages

$$\frac{1}{N}\phi(\boldsymbol{h}^{(\ell)}_{\mu}(t)) \cdot \phi(\boldsymbol{h}^{(\ell)}_{\alpha}(s)) \longrightarrow \left\langle \phi(h^{(\ell)}_{\mu}(t))\phi(h^{(\ell)}_{\mu}(s)) \right\rangle$$

Gradient-flow in feature space

Gradient-flow can be rewritten in “feature/activation” space *without reference to weights except at initialization*. Key quantities are layer-wise feature and gradient kernels.

$$\Phi^{(\ell)}_{\mu\nu}(t,s) = \frac{1}{N}\phi(\mathbf{h}^{(\ell)}_{\mu}(t)) \cdot \phi(\mathbf{h}^{(\ell)}_{\nu}(s)) \ , \qquad G^{(\ell)}_{\mu\nu}(t,s) = \frac{1}{N}\mathbf{g}^{(\ell)}_{\mu}(t) \cdot \mathbf{g}^{(\ell)}_{\nu}(s) \qquad \mathbf{g}^{(\ell)}_{\mu} \equiv \sqrt{N}\frac{\partial h^{(L)}_{\mu}}{\partial \mathbf{h}^{(\ell)}_{\mu}}$$

$$\frac{\partial f_\mu}{\partial t} = \frac{\eta_0}{P}\sum_\nu \Delta_\nu(t)\left[\Phi^{(L-1)}_{\mu\nu} + \sum_{\ell=1}^{L-1} G^{(\ell)}_{\mu\nu}\Phi^{(\ell-1)}_{\mu\nu}\right] \qquad \Delta_\mu = -\frac{\partial \mathcal{L}}{\partial f_\mu}$$

Dynamical Mean Field Theory (DMFT)

$$\{u_\mu^\ell(t)\}_{\mu\in[P],t\in\mathbb{R}_+} \sim \mathcal{GP}(0,\boldsymbol{\Phi}^{\ell-1})\ ,\ \{r_\mu^\ell(t)\}_{\mu\in[P],t\in\mathbb{R}_+} \sim \mathcal{GP}(0,\boldsymbol{G}^{\ell+1}),$$

$$h_\mu^\ell(t) = u_\mu^\ell(t) + \gamma_0 \int_0^t ds \sum_{\alpha=1}^P \left[A_{\mu\alpha}^{\ell-1}(t,s) + \Delta_\alpha(s)\Phi_{\mu\alpha}^{\ell-1}(t,s)\right] z_\alpha^\ell(s)\dot\phi(h_\alpha^\ell(s)),$$

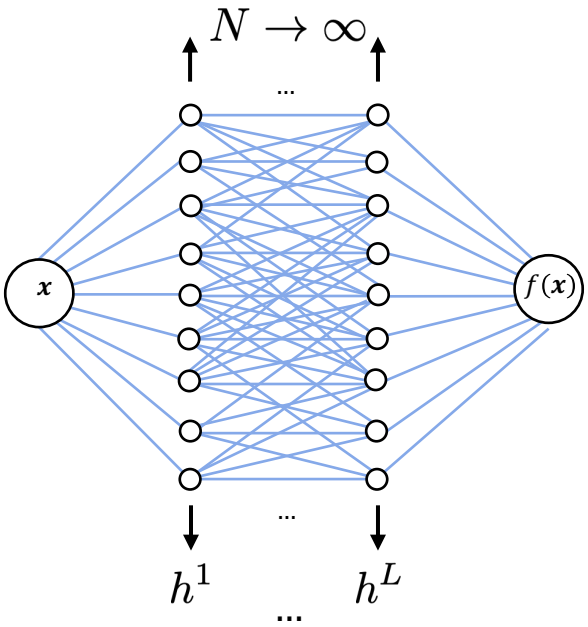
$$z_\mu^\ell(t) = r_\mu^\ell(t) + \gamma_0 \int_0^t ds \sum_{\alpha=1}^P \left[B_{\mu\alpha}^{\ell}(t,s) + \Delta_\alpha(s)G_{\mu\alpha}^{\ell+1}(t,s)\right] \phi(h_\alpha^\ell(s)),$$

$$g_\mu^\ell(t) = \dot\phi(h_\mu^\ell(t))z_\mu^\ell(t)$$

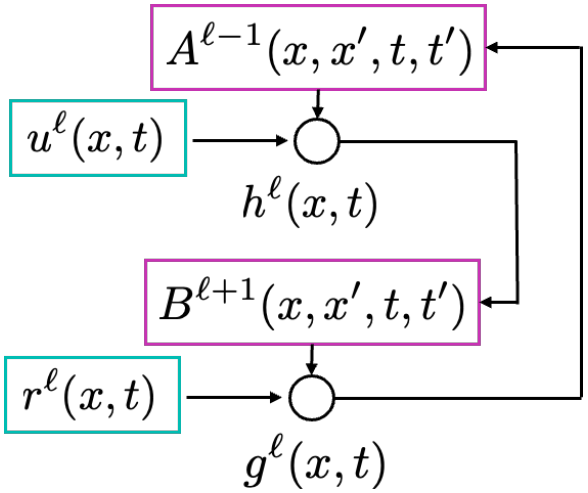
$$\Phi_{\mu\alpha}^\ell(t,s) = \left\langle \phi(h_\mu^\ell(t))\phi(h_\alpha^\ell(s))\right\rangle\ ,\ G_{\mu\alpha}^\ell(t,s) = \left\langle g_\mu^\ell(t) g_\alpha^\ell(s)\right\rangle$$

$$A_{\mu\alpha}^\ell(t,s) = \gamma_0^{-1}\left\langle \frac{\delta\phi(h_\mu^\ell(t))}{\delta r_\alpha^\ell(s)}\right\rangle\ ,\ B_{\mu\alpha}^\ell(t,s) = \gamma_0^{-1}\left\langle \frac{\delta g_\mu^{\ell+1}(t)}{\delta u_\alpha^{\ell+1}(s)}\right\rangle$$

Recovers Yang & Hu 2020 results from Tensor Programs in relevant limits/settings

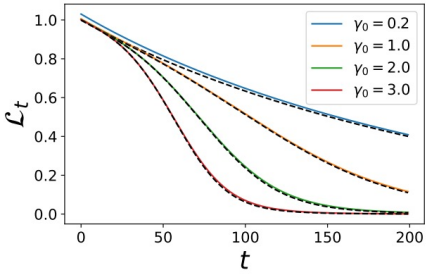


$$\sim \quad N \to \infty$$

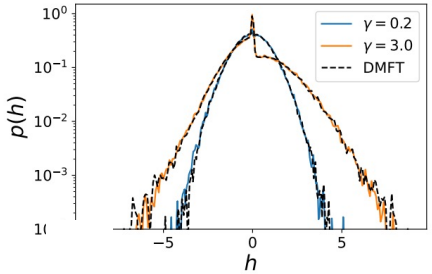


DMFT matches experiments

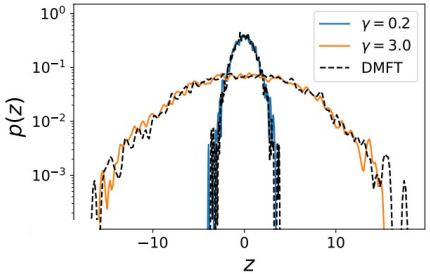
Richness: Infinite width equations depend crucially on an output multiplier γ_0



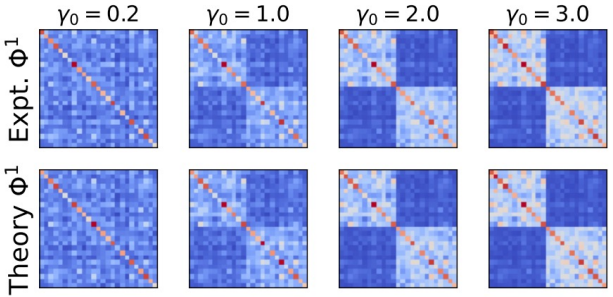
(a) Loss Dynamics



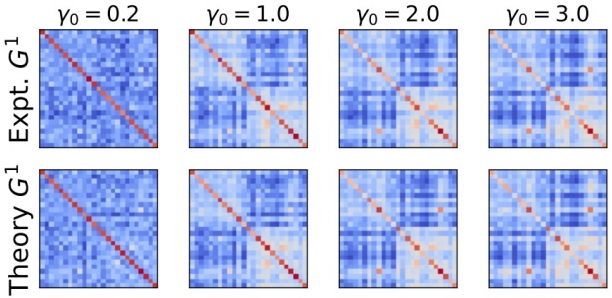
(b) Final h Distribution



(c) Final z Distribution



(d) Final Φ^1 Kernels



(e) Final G^1 Kernels

(Depth 4, Width 4000, tanh, synthetic data)

Comments

1. For two-layer networks, A and B field (response functions) disappear. One can derive a PDE version of the dynamics. However, unlike previous PDE descriptions in this limit (Rotskoff, Vanden-Eijnden 2018; Mei, Nguyen, Montanari 2018) (see Andrea Montanari's Turing Lecture 3), the PDE describes the density h and z, not weights

2. For linear networks, DMFT closes under kernels to give deterministic, algebraic equations alleviating the need for a Monte Carlo procedure. Two-layer version can be solved exactly recovering known results in linear networks (e.g. Saxe et al 2013).

3. A DMFT can be given for SGD. One does this by first committing to a minibatch B_t at each gradient step. Integrals over time become sums over discrete time and the data sum is over B_t.

Limiting process for a residual network through the DMFT (Informal)

$$\mathbf{h}^{\ell+1} = \mathbf{h}^{\ell} + \frac{1}{L^{1/2}} \mathbf{W}^{\ell} \phi(\mathbf{h}^{\ell})$$

Take depth (L) and width (N) to infinity in any sequential limit $\tau = \frac{\ell}{L} \in [0, 1]$

$$h(\tau; \boldsymbol{x}; t) = h(0; \boldsymbol{x}; t) + \int_0^{\tau} du(\tau'; \boldsymbol{x}; t) + \eta_0 \gamma_0 \int_0^{\tau} d\tau' \int_0^{t} ds \int d\boldsymbol{x}' C_h(\tau'; \boldsymbol{x}, \boldsymbol{x}'; t, s) g(\tau'; \boldsymbol{x}'; s)$$

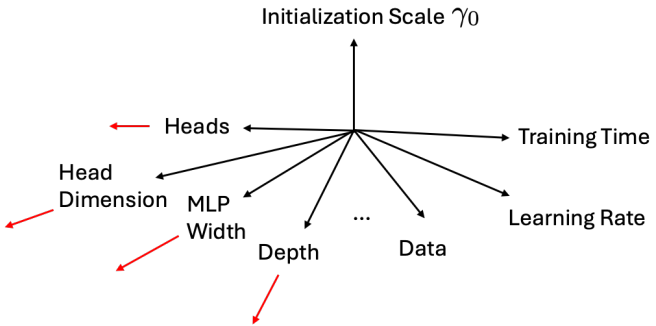
Full characterization of the deterministic operator C can be found in the paper

Bordelon et al., 2023;

Infinite Limits of Multi-head Transformer Dynamics

Blake Bordelon, Hamza Chaudhry, Cengiz Pehlevan
John A. Paulson School of Engineering and Applied Sciences
Center for Brain Science
Kempner Institute for the Study of Natural and Artificial Intelligence
Harvard University
Cambridge, MA 02138
blake_bordelon@g.harvard.edu
hchaudhry@g.harvard.edu
cpehlevan@seas.harvard.edu

NeurIPS 2024



What does this theory imply for scaling laws?

Scaling exponents

Kaplan et al., 2020

$$\mathcal{L}(N,T) = \left[\left(\frac{N_c}{N}\right)^{\alpha_N/\alpha_T} + \frac{T_c}{T}\right]^{\alpha_T} \qquad \alpha_N = 0.076, \qquad \alpha_T = 0.095$$

Hoffman et al., 2022

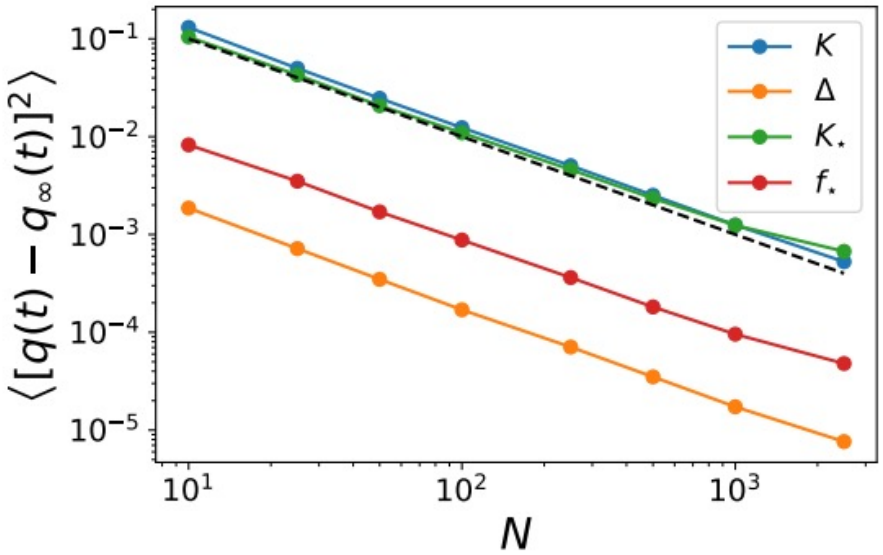
$$\mathcal{L}(N,T) = E + N^{-\alpha_N} + T^{-\alpha_T} \qquad \alpha_N = 0.39, \qquad \alpha_T = 0.28$$

Compute proportional to NT

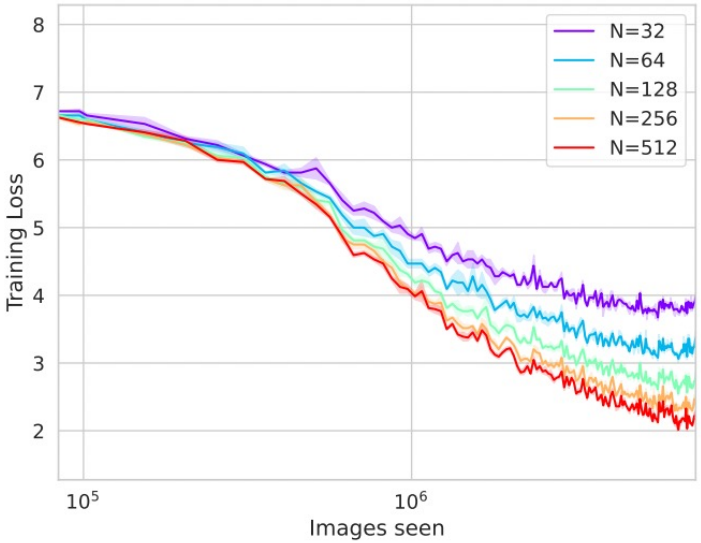
Dynamics of Finite Width Kernel and Prediction Fluctuations in Mean Field Neural Networks

Blake Bordelon & Cengiz Pehlevan

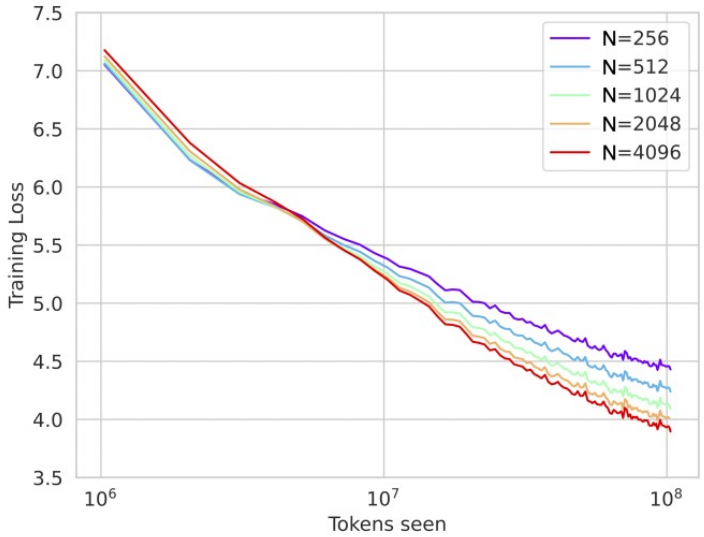
NeurIPS 2023



Deviations from consistent behavior across widths for more complex datasets



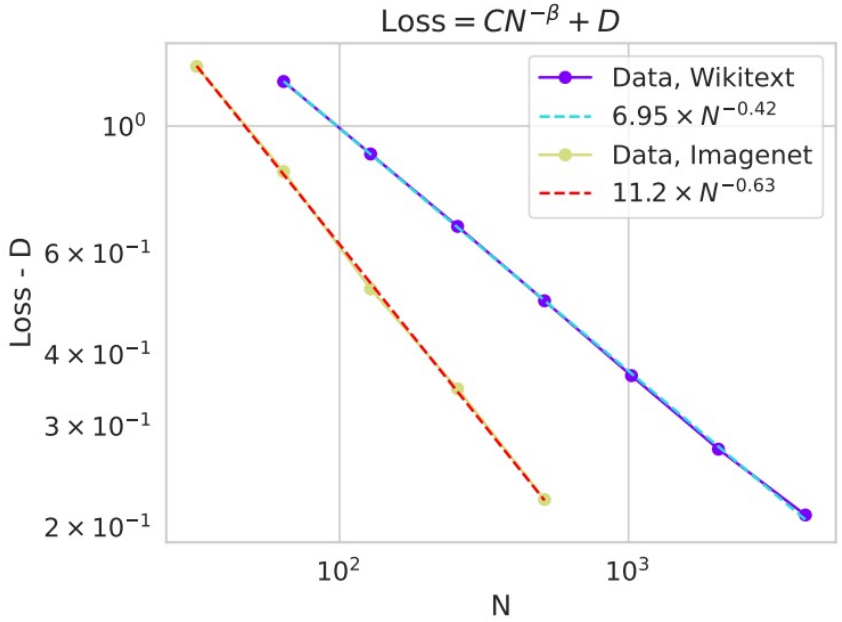
(b) Imagenet



(c) Wikitext-103

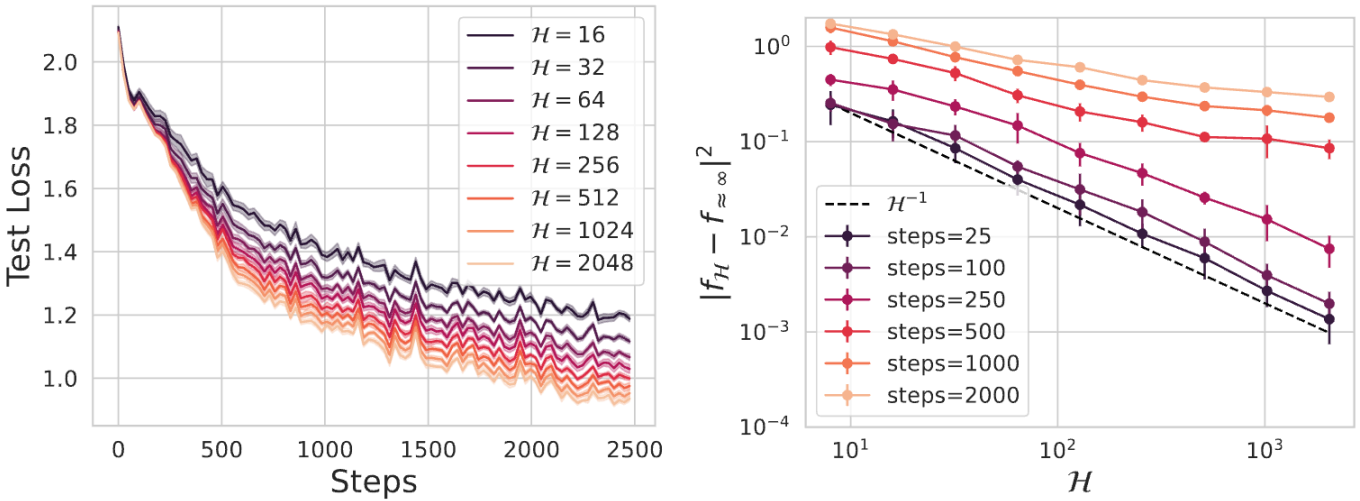
More experiments in Nikhil Vyas*, Alex Atanasov*, Blake Bordelon*, Depen Morwani, Sab Sainathan, Cengiz Pehlevan, NeurIPS, 2023

Late time loss scaling with parameters deviates from 1/width predicted by DMFT



Deviations from consistent behavior across widths

SGD dynamics for a vision transformer trained on CIFAR-5M. H – number of attention heads



(a) Training Dynamics Varying $\mathcal{H}$

(b) Convergence to $\mathcal{H} \to \infty$ limit

Bordelon, Chaudhry, Pehlevan

Infinite Limits of Multi-head Transformer Dynamics (NeurIPS 2024)

Part III: Scaling Laws

A Dynamical Model of Neural Scaling Laws

Blake Bordelon [1 2] Alexander Atanasov [3 2] Cengiz Pehlevan [1 2]

ICML 2024

How Feature Learning Can Improve Neural Scaling Laws

Blake Bordelon*, Alexander Atanasov* , Cengiz Pehlevan

ICLR 2025

- Related “static” works by: Caponnetto & De Vito 2007; Bordelon et al., 2020; Spigler et al., 2020; Bahri et al. 2021; Mel & Ganguli 2021; Favero et al. 2021; Maloney et al. 2022; Atanasov et al. 2022; Cui et al. 2022; Cagnetta et al. 2023; Simon et al. 2023; Dohmatob et al. 2024; Defilippis et al. 2024
- More recent “dynamic” works: Paquette et al. 2024; Lin et al. 2024
- Scaling laws in the feature learning regime: Worschech & Rosenow 2025
- Other approaches: Michaud et al. 2023;….

Resources

LECTURE NOTES, REVIEWS AND BLOG POSTS

Replica Method for the Machine Learning Theorist - Part 1, by Blake Bordelon, Haozhe Shan, Abdul Canatar, Boaz Barak, Cengiz Pehlevan

Replica Method for the Machine Learning Theorist - Part 2, by Blake Bordelon, Haozhe Shan, Abdul Canatar, Boaz Barak, Cengiz Pehlevan

Lecture notes on the replica method for Wishart matrix eigenvalues, by Jacob Zavatone-Veth

A brief introduction to the neural network Gaussian process from the perspective of mean field theory, by Jacob Zavatone-Veth

Lecture Notes on Infinite-Width Limits of Neural Networks, Cengiz Pehlevan and Blake Bordelon, prepared for 2023 Princeton ML Theory Summer School

Infinite Limits of Neural Networks, Deeper Learning Blog, by Alex Atanasov, Blake Bordelon and Cengiz Pehlevan

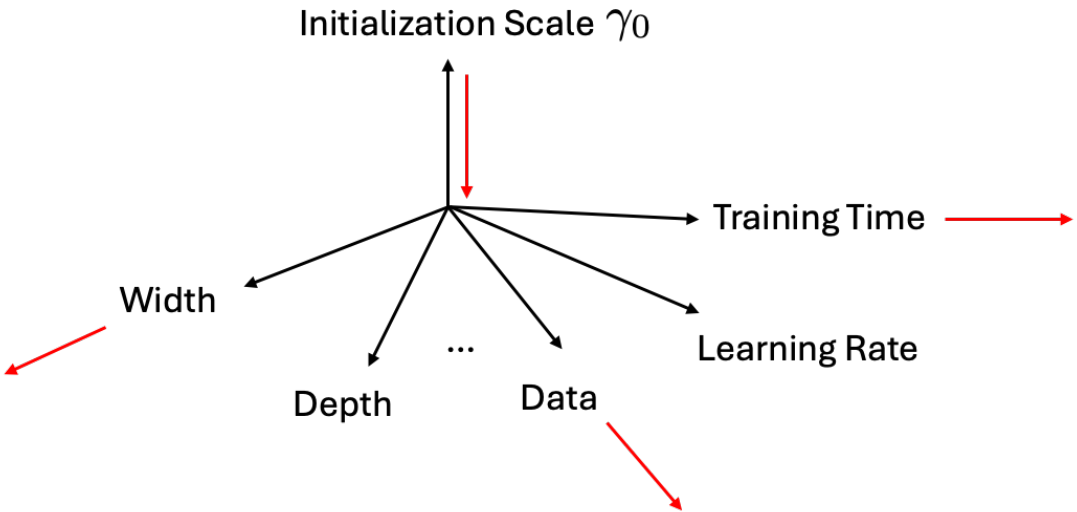
A Dynamical Model of Neural Scaling Laws, Deeper Learning Blog, by Blake Bordelon, Alex Atanasov and Cengiz Pehlevan

Scaling and renormalization in high-dimensional regression, by Alex Atanasov, Jacob Zavatone-Veth, Cengiz Pehlevan

Solvable Model of In-Context Learning Using Linear Attention, Deeper Learning Blog, by Mary Letey

We need to also scale data size!

Typical case behavior of this limit over random draws of data and initial parameters



This simple theory captures how the behavior of the model depends on computational/statistical resources (width, training time, and total data)

Main ideas of the model

- We are looking for a simple model where we can vary parameters (N), dataset size (P) and training time (T)

- For analytical tractability, we will consider the lazy limit of neural network training. (I will later generalize to feature learning).

- In the lazy limit, neural networks are kernel machines (NTK in the infinite width, Jacot et al. 2018)

- Main modeling idea: Kernels of finite-width networks are “noisy” versions of the infinite-width models.

Bahri et al. 2021, Atanasov et al. 2022, Maloney et al. 2022

Teacher-Student Setup

Eigenvalues and eigenfunctions of the infinite-width NTK are a complete basis:

$$K_\infty(\mathbf{x}, \mathbf{x}') = \sum_k \psi_k^\infty(\mathbf{x})\psi_k^\infty(\mathbf{x}') \qquad \int d\mathbf{x}\, p(\mathbf{x})\psi_k^\infty(\mathbf{x})\psi_l^\infty(\mathbf{x}) = \lambda_k \delta_{kl}$$

Finite-width model: $f(\mathbf{x}) = \sum_k w_k \psi_k^N(\mathbf{x})$

(student, width N)

Expand: $\psi_k^N(\mathbf{x}) = \frac{1}{\sqrt{N}} \sum_l A_{kl}\psi_l^\infty(\mathbf{x})$

$$\lim_{N\to\infty} \frac{1}{N}\mathbf{A}^\top \mathbf{A} = \mathbf{I} \qquad A_{ij} \sim \mathcal{N}(0,1)$$

Teacher: $y(\mathbf{x}) = \sum_k \bar{w}_k \psi_k^\infty(\mathbf{x})$

Data: $\mathcal{D} = (\mathbf{x}_\mu, y_\mu)_{\mu=1}^P \qquad \mathbf{x}_\mu \sim p(\mathbf{x}) \qquad y_\mu = y(\mathbf{x}_\mu) + \epsilon_\mu \qquad \Psi_{\mu k} \equiv \psi_k^\infty(\mathbf{x}_\mu)$

Teacher-Student Setup

Teacher: $y(x) = \sum_k \bar{w}_k \psi_k^\infty$

Student: $f(x) = \sum_l \frac{1}{\sqrt{N}} \sum_k w_k A_{kl} \psi_l^\infty(x)$

Consider gradient-flow on MSE loss

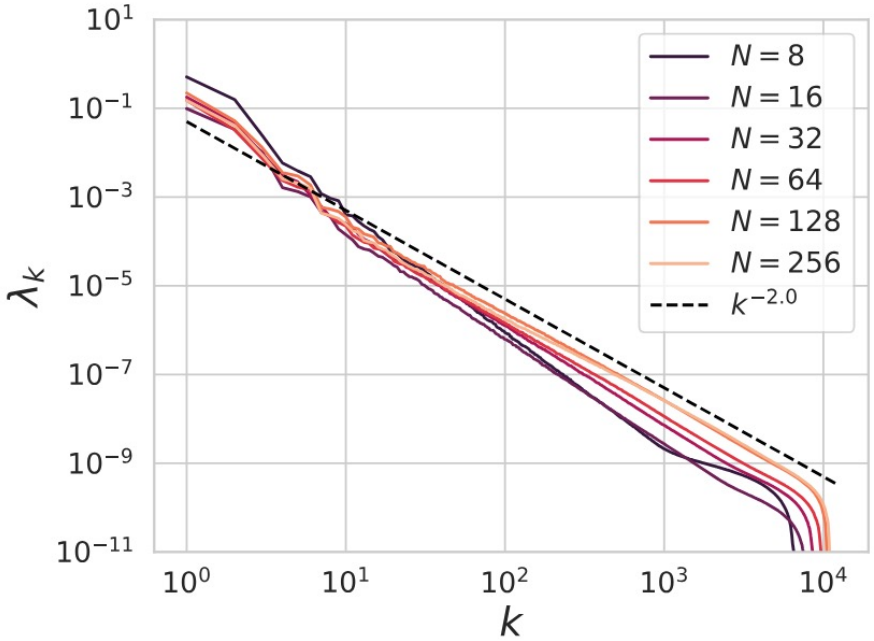
$$v_k^0 \equiv \bar{w}_k - \frac{1}{\sqrt{N}} \sum_{l=1}^{N} A_{lk} w_l$$

$$\frac{d}{dt}\mathbf{v}^0(t) = -\left(\frac{1}{N}\mathbf{A}^\top \mathbf{A}\right)\left(\frac{1}{P}\mathbf{\Psi}^\top \mathbf{\Psi}\right)\mathbf{v}^0(t)$$

- Using path integral (MSR) methods and Gaussian equivalence, we study the $P, N >> 1$ regime where we took $M \to \infty$ first. (Technical note: Results are correct up to $\mathcal{O}(1/P + 1/N)$. There is an asymptotically exact limit as well with structurally identical equations where $N/M = \nu$, $P/M = \alpha$, $M, N, P \to \infty$.)
- Averages over two types of disorder: initialization and data.
- We derive a mean field theory for full asymptotic learning dynamics and study other phenomena as well. See paper for these equations and results.

Data has power-law structure

ResNets on CIFAR-5M



(a) NTK Spectra for Varying Widths

Power law in – power law out

$$K_\infty(\mathbf{x},\mathbf{x}') = \sum_k \psi_k^\infty(\mathbf{x})\psi_k^\infty(\mathbf{x}') \qquad \int d\mathbf{x}\, p(\mathbf{x})\psi_k^\infty(\mathbf{x})\psi_l^\infty(\mathbf{x}) = \lambda_k \delta_{kl} \qquad y(x) = \sum_k \bar{w}_k \psi_k^\infty$$

Source-and-capacity constraints: $(\bar{w}_k)^2 \lambda_k \sim k^{-a}, \quad \lambda_k \sim k^{-b}$

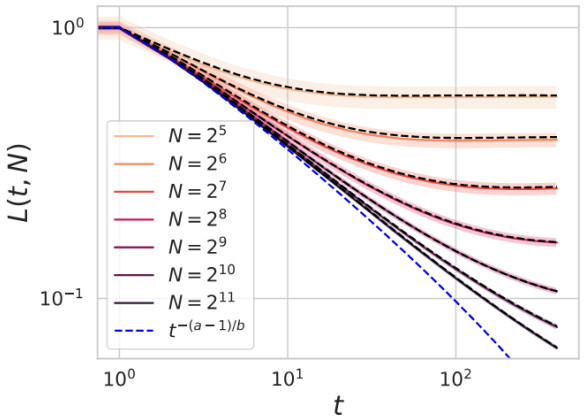
$$\mathcal{L}(t,P,N) \approx \begin{cases} t^{-(a-1)/b}\ ,\ P,N \to \infty\ ,\ \text{(Time-Bottleneck)} \\ P^{-(a-1)}\ ,\ t,N \to \infty\ ,\ \text{(Data-Bottleneck)} \\ N^{-(a-1)}\ ,\ t,P \to \infty\ ,\ \text{(Model-Bottleneck)} \end{cases}$$

Caponnetto & De Vito 2007;
Bordelon et al., 2020;
Spigler et al., 2020;
Bahri et al. 2021;
Favero et al. 2021;
Maloney et al. 2022;
Cui et al. 2022;
Cagnetta et al. 2023;
Simon et al. 2023;
Dohmatob et al. 2024;
Defilippis et al.; 2024
…

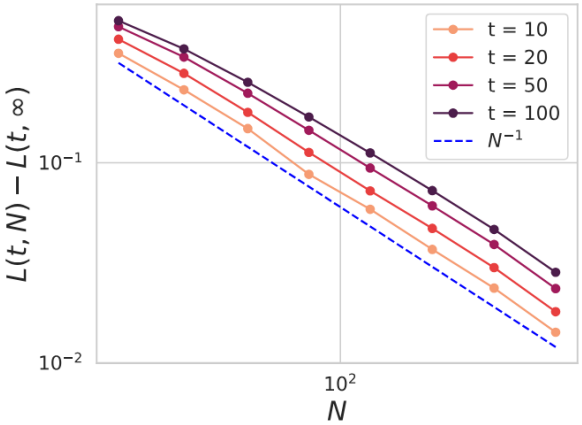
$$(\bar{w}_k)^2 \lambda_k \sim k^{-a}, \quad \lambda_k \sim k^{-b}$$

$$\mathcal{L}(t, P, N) \approx \begin{cases} t^{-(a-1)/b} \ , \ P, N \to \infty \ , \ \text{(Time-Bottleneck)} \\ P^{-(a-1)} \ , \ t, N \to \infty \ , \ \text{(Data-Bottleneck)} \\ N^{-(a-1)} \ , \ t, P \to \infty \ , \ \text{(Model-Bottleneck)} \end{cases}$$

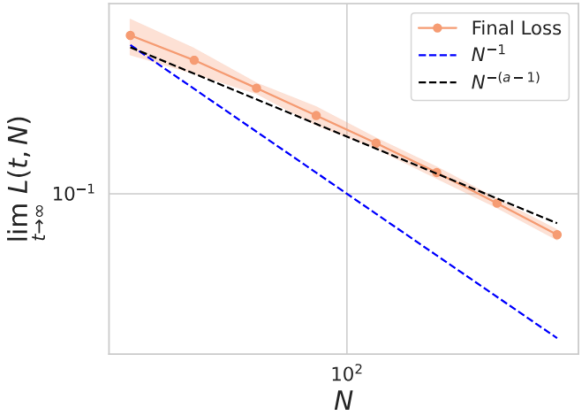
$$a = 1.5, \quad b = 1.25$$



(a) $P = 1000$ Test Loss Dynamics



(b) Early Time Model Convergence

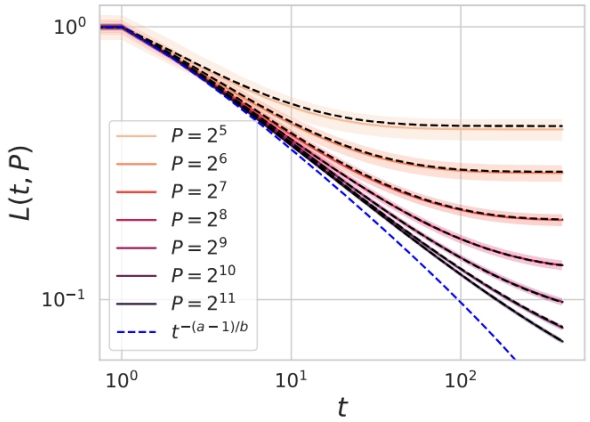


(c) Late Time Model Bottleneck

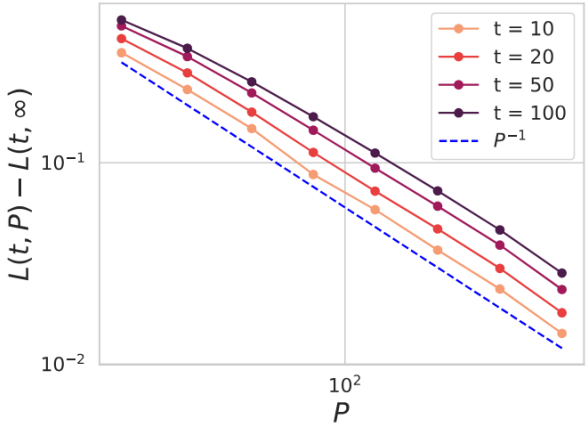
$$(\bar{w}_k)^2 \lambda_k \sim k^{-a}, \quad \lambda_k \sim k^{-b}$$

$$\mathcal{L}(t, P, N) \approx \begin{cases} t^{-(a-1)/b} \ , \ P, N \to \infty \ , \ \text{(Time-Bottleneck)} \\ P^{-(a-1)} \ , \ t, N \to \infty \ , \ \text{(Data-Bottleneck)} \\ N^{-(a-1)} \ , \ t, P \to \infty \ , \ \text{(Model-Bottleneck)} \end{cases}$$

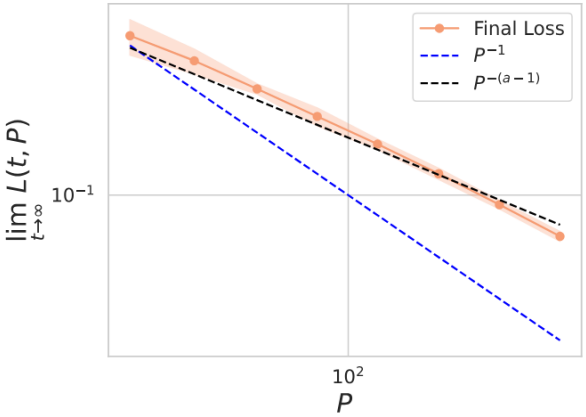
$$a = 1.5, \quad b = 1.25$$



(d) $N = 1000$ Test Loss Dynamics



(e) Early Time Data Convergence



(f) Late Time Data Bottleneck

Bottlenecks as rank-constraints

$$\mathcal{L}(t,P,N) \approx \begin{cases} t^{-(a-1)/b} \ , \ P,N \to \infty \ , \ \text{(Time-Bottleneck)} \\ P^{-(a-1)} \ , \ t,N \to \infty \ , \ \text{(Data-Bottleneck)} \\ N^{-(a-1)} \ , \ t,P \to \infty \ , \ \text{(Model-Bottleneck)} \end{cases}$$

$$\mathcal{L} \approx \sum_{k>k_\star} (\bar{w}_k)^2 \lambda_k \approx k_\star^{-(a-1)}$$

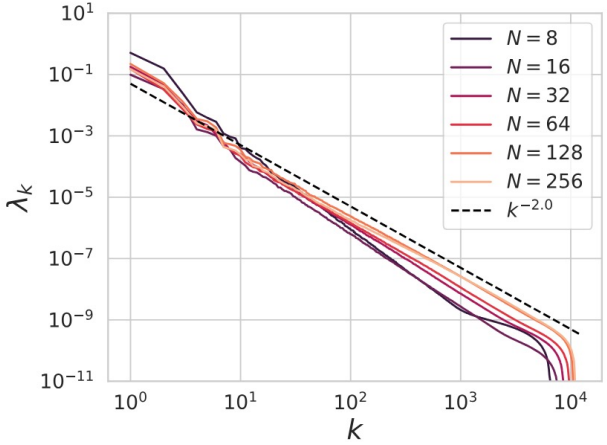
Data Bottleneck: $k_\star = P$

Model Bottleneck: $k_\star = N$

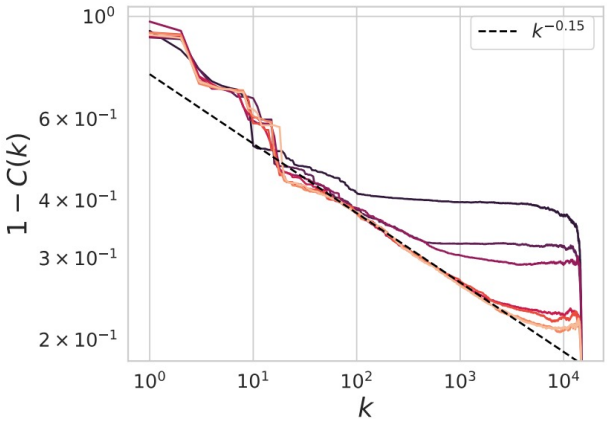
Time Bottleneck: $k_\star = t^{1/b}$ $\qquad (\lambda_k = k^{-b}, \tau_k \sim k^b)$

Spigler et al., 2020

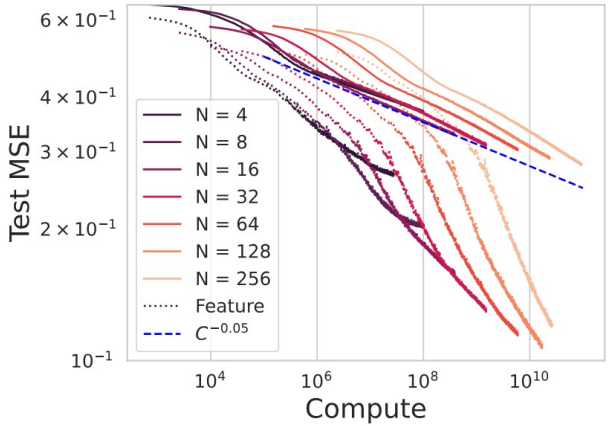
ResNets on CIFAR-5M



(a) NTK Spectra for Varying Widths



(b) Task-Power Decay



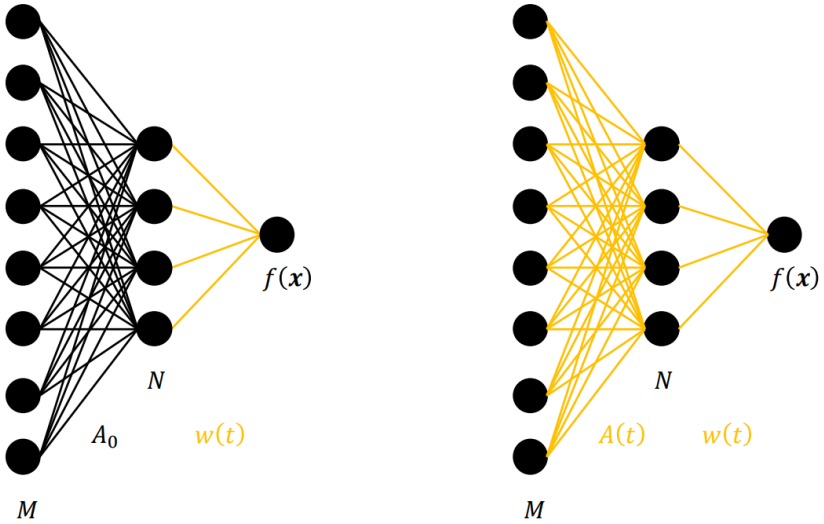
(c) Lazy vs Feature Compute Scaling

$$C(k) \equiv \frac{\sum_{i \leq k} \lambda_i (\bar{w}_i)^2}{\sum_i \lambda_i (\bar{w}_i)^2}$$

How Feature Learning Can Improve Neural Scaling Laws

Blake Bordelon*, Alexander Atanasov* , Cengiz Pehlevan

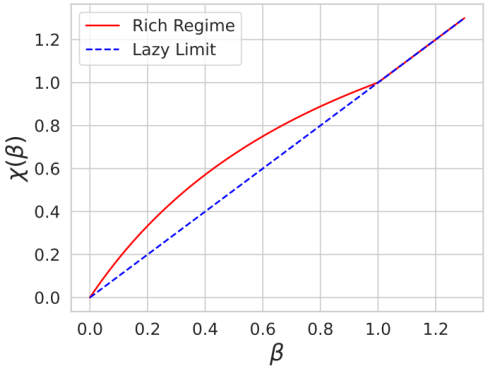
ICLR 2025



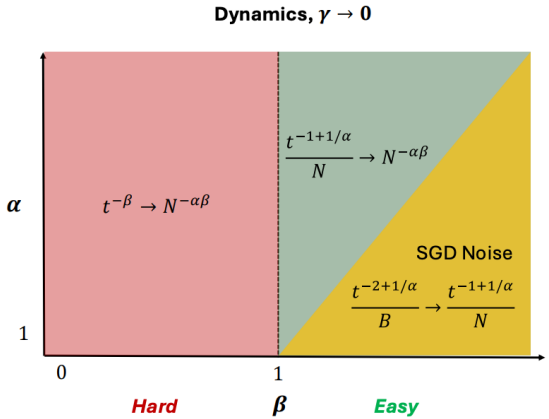
$$M \to \infty, \quad N, B >> 1$$

Improved scaling for hard, but not easy tasks from feature learning

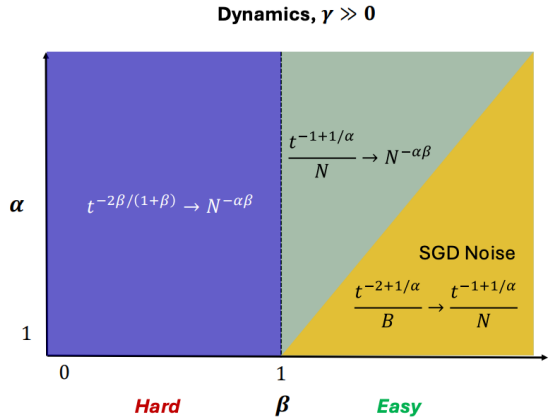
$$\lambda_k \sim k^{-\alpha}\,, \qquad \sum_{\ell>k} \lambda_\ell (w_\ell^*)^2 \sim \lambda_k^{-\beta} \sim k^{-\alpha\beta}$$



(a) $\lim_{N\to\infty} \mathcal{L}(t, N) \sim t^{-\chi(\beta)}$



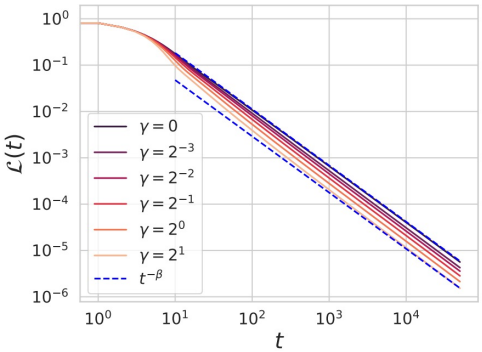
(b) Lazy Limit



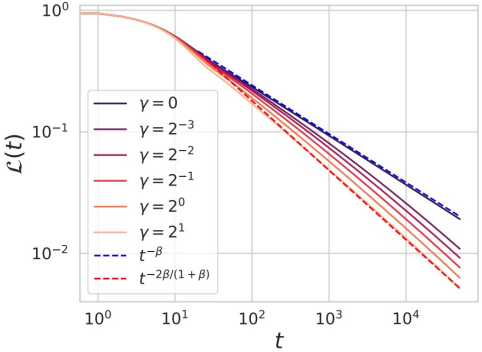
(c) Rich Regime

Hard task: out of RKHS

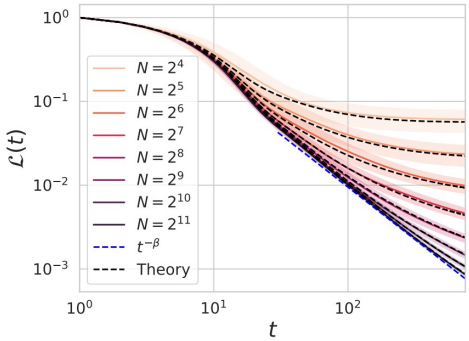
$$|y|_{\mathcal{H}}^2 = \sum_k (w_k^\star)^2 = \sum_k k^{-\alpha(\beta-1)-1} \approx \begin{cases} \frac{1}{\alpha(\beta-1)} & \beta > 1 \\ \infty & \beta < 1 \end{cases}$$



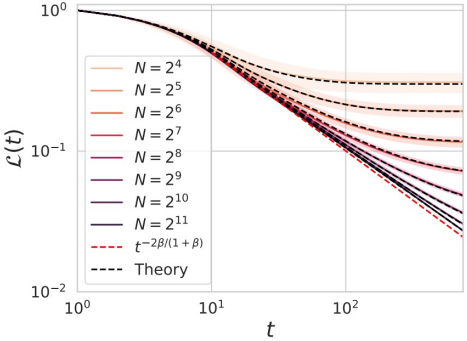
(a) Easy Task $\beta = 1.2$ with $N, B \to \infty$



(b) Hard Task $\beta = 0.4$ with $N, B \to \infty$



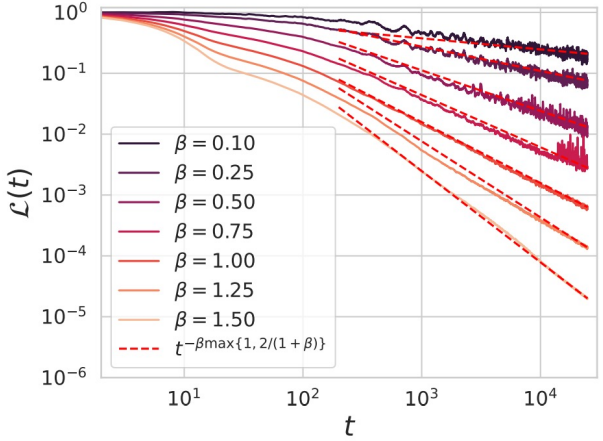
(c) Easy Task $\beta = 1.2$ with finite N



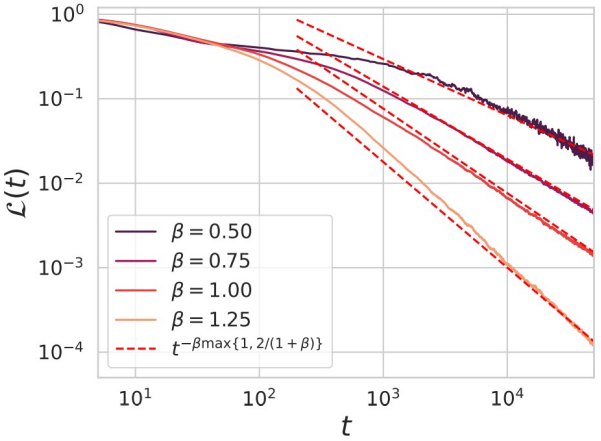
(d) Hard Task $\beta = 0.4$ with finite N

Theory predicts correct power exponents for deep ReLU networks

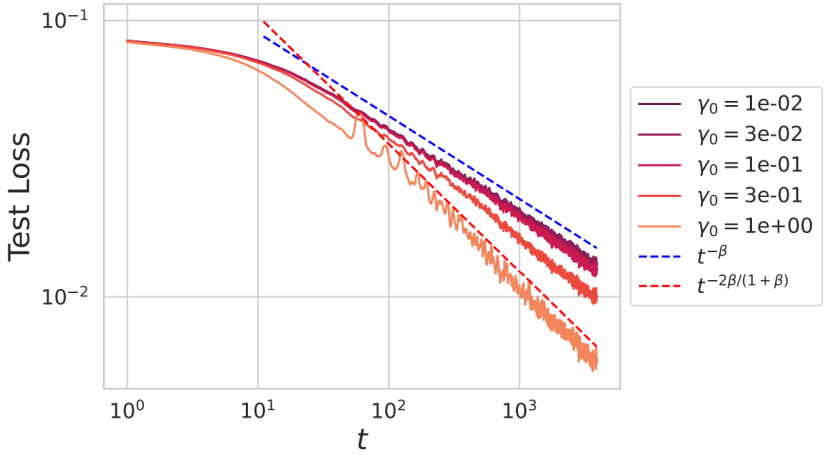
$$y(\theta) = \sum_{k=1}^{\infty} k^{-q} \cos(k\theta)\ ,\ K(\theta, \theta') = \sum_{k=1}^{\infty} \lambda_k \cos(k(\theta - \theta'))$$



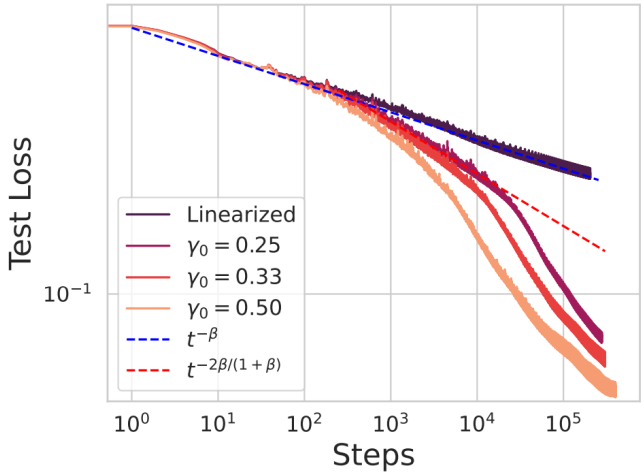
(a) ReLU ($q_\phi = 1.0$) with varying $\beta = \frac{2q-1}{2q_\phi}$



(b) Varying q_ϕ with fixed target ($q = 1.4$)



(a) CNNs on MNIST-1M, $\beta = 0.30$



(b) CNNs on CIFAR-5M $\beta = 0.075$

Summary from these models

- Data structure and architecture jointly determine scaling exponents
- Feature learning may not always improve scaling exponents
- Models may transition between different scaling regimes during training

Acknowledgments

Adam Lee
Alex Meterez
Blake Bordelon
Hamza Chaudhry
Indranil Halder
Ganesh Kumar
Clarissa Lauditi
Adam Lee
Mary Letey
Mo Osman
Billy Qian
Ben Ruben
Sabarish Sainathan
William Tong
Ningjing Xia

Alumni:
Abdulkadir Canatar
Alex Atanasov
Jacob Zavatone-Veth
Anindita Maiti
Nikhil Vyas

Collaborators:
Boris Hanin
Mufan Li
Lorenzo Noci
Depen Morwani
CEREBRAS

HDSI
NIH
intel
NSF
Google
DARPA
ALFRED P. SLOAN FOUNDATION

Harvard John A. Paulson School of Engineering and Applied Sciences

Kempner Institute FOR THE STUDY OF NATURAL & ARTIFICIAL INTELLIGENCE