Analyzing Finite Neural Networks: Can We Trust Neural Tangent Kernel Theory?

Mariia Seleznova & Gitta Kutyniok

(Ludwig-Maximilians-Universität München)

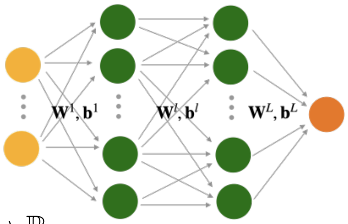
MSML 2021

August 16-19, 2021

Deep Neural Networks (DNNs)

Definition: Assume the following notation:

- Number of layers $L \geq 2$.
- Layers' widths M_ℓ, $\ell = 0, \ldots, L$.
- *Weights* $W^\ell \in \mathbb{R}^{M_\ell \times M_{\ell-1}}$, $\ell \geq 1$.
- *Biases* $b^\ell \in \mathbb{R}^{M_l \ell}$, $\ell \geq 1$.
- (Non-linear) *activation function* $\phi : \mathbb{R} \to \mathbb{R}$.



Then a *deep neural network (DNN)* is a function $f : \mathbb{R}^{M_0} \to \mathbb{R}^{M^L}$:

$$f(x) = \mathrm{W}^L \phi(\mathrm{W}^{L-1} \phi(\mathrm{W}^{L-2} \phi(\mathrm{W}^{L-3} \ldots) + b^{L-1}) + b^L.$$

DNNs' training dynamics

Consider training a DNN with parameters $\theta = \{(W^\ell, b^\ell)\}_{\ell=1,\dots,L}$ on dataset $D = (X, Y)$, $X \in \mathbb{R}^{N \times M_0}$, $Y \in \mathbb{R}^{N \times M_L}$ by *gradient flow* in time t with *loss function* $\mathcal{L}$:

$$\dot{\theta}^{(t)} = -\nabla_\theta \mathcal{L}(f^{(t)}(X), Y)$$

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

DNNs' training dynamics

Consider training a DNN with parameters $\theta = \{(W^\ell, b^\ell)\}_{\ell=1,\dots,L}$ on dataset $D = (X, Y)$, $X \in \mathbb{R}^{N \times M_0}$, $Y \in \mathbb{R}^{N \times M_L}$ by *gradient flow* in time t with *loss function* $\mathcal{L}$:

$$\dot{\theta}^{(t)} = -\nabla_\theta \mathcal{L}(f^{(t)}(X), Y)$$

Then the dynamics of the output function on any input $\tilde{X} \in \mathbb{R}^{\tilde{N} \times M_0}$ is given by:

$$\dot{f}^{(t)}(\tilde{X}) = \nabla_\theta f^{(t)}(\tilde{X}) \cdot \dot{\theta}^{(t)}$$

DNNs' training dynamics

Consider training a DNN with parameters $\theta = \{(W^\ell, b^\ell)\}_{\ell=1,\dots,L}$ on dataset $D = (X, Y)$, $X \in \mathbb{R}^{N \times M_0}$, $Y \in \mathbb{R}^{N \times M_L}$ by *gradient flow* in time t with *loss function* $\mathcal{L}$:

$$\dot{\theta}^{(t)} = -\nabla_\theta \mathcal{L}(f^{(t)}(X), Y)$$

Then the dynamics of the output function on any input $\tilde{X} \in \mathbb{R}^{\tilde{N} \times M_0}$ is given by:

$$\dot{f}^{(t)}(\tilde{X}) = \nabla_\theta f^{(t)}(\tilde{X}) \cdot \dot{\theta}^{(t)}$$

Challenges:

- No analytical solutions for $f^{(t)}$ in general.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

DNNs' training dynamics

Consider training a DNN with parameters $\theta = \{(W^\ell, b^\ell)\}_{\ell=1,\dots,L}$ on dataset $D = (X, Y)$, $X \in \mathbb{R}^{N \times M_0}$, $Y \in \mathbb{R}^{N \times M_L}$ by *gradient flow* in time t with *loss function* $\mathcal{L}$:

$$\dot{\theta}^{(t)} = -\nabla_\theta \mathcal{L}(f^{(t)}(X), Y)$$

Then the dynamics of the output function on any input $\tilde{X} \in \mathbb{R}^{\tilde{N} \times M_0}$ is given by:

$$\dot{f}^{(t)}(\tilde{X}) = \nabla_\theta f^{(t)}(\tilde{X}) \cdot \dot{\theta}^{(t)}$$

Challenges:

- No analytical solutions for $f^{(t)}$ in general.
- No access to *generalization error* $\mathbb{E}_{x,y}[\mathcal{L}(f^{(t)}(x), y)]$.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

DNNs' training dynamics

Consider training a DNN with parameters $\theta = \{(W^\ell, b^\ell)\}_{\ell=1,\ldots,L}$ on dataset $D = (X, Y)$, $X \in \mathbb{R}^{N\times M_0}$, $Y \in \mathbb{R}^{N\times M_L}$ by *gradient flow* in time t with *loss function* $\mathcal{L}$:

$$\dot{\theta}^{(t)} = -\nabla_\theta \mathcal{L}(f^{(t)}(X), Y)$$

Then the dynamics of the output function on any input $\tilde{X} \in \mathbb{R}^{\tilde{N}\times M_0}$ is given by:

$$\dot{f}^{(t)}(\tilde{X}) = \nabla_\theta f^{(t)}(\tilde{X}) \cdot \dot{\theta}^{(t)}$$

Challenges:

- No analytical solutions for $f^{(t)}$ in general.
- No access to *generalization error* $\mathbb{E}_{x,y}[\mathcal{L}(f^{(t)}(x), y)]$.
- No access to model's stability and robustness.

DNNs' training dynamics

Consider training a DNN with parameters $\theta = \{(W^\ell, b^\ell)\}_{\ell=1,\dots,L}$ on dataset $D = (X, Y)$, $X \in \mathbb{R}^{N \times M_0}$, $Y \in \mathbb{R}^{N \times M_L}$ by *gradient flow* in time t with *loss function* $\mathcal{L}$:

$$\dot{\theta}^{(t)} = -\nabla_\theta \mathcal{L}(f^{(t)}(X), Y)$$

Then the dynamics of the output function on any input $\tilde{X} \in \mathbb{R}^{\tilde{N} \times M_0}$ is given by:

$$\dot{f}^{(t)}(\tilde{X}) = \nabla_\theta f^{(t)}(\tilde{X}) \cdot \dot{\theta}^{(t)}$$

Challenges:

- No analytical solutions for $f^{(t)}$ in general.
- No access to *generalization error* $\mathbb{E}_{x,y}[\mathcal{L}(f^{(t)}(x), y)]$.
- No access to model's stability and robustness.

$\rightsquigarrow$ *Neural Tangent Kernel (NTK) theory addresses these challenges in a special case of infinitely-wide DNNs!*

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Neural Tangent Kernel Theory

Consider *squared loss* $\mathcal{L}(\hat{Y}, Y) = \frac{1}{2N}\|(\hat{Y} - Y)\|_2^2$ and for simplicity set $M_L = 1$. Then the gradient flow dynamics of a DNN takes form:

$$\dot{\theta}^{(t)} = -\nabla_\theta \mathcal{L}(f^{(t)}(X), Y) = -\frac{1}{N}\nabla_\theta f^{(t)}(X)^T \cdot \big(f^{(t)}(X) - Y\big),$$

$$\dot{f}^{(t)}(\tilde{X}) = \nabla_\theta f^{(t)}(\tilde{X}) \cdot \dot{\theta}^{(t)} = -\frac{1}{N}\underbrace{\nabla_\theta f^{(t)}(\tilde{X})\nabla_\theta f^{(t)}(X)^T}_{\Theta^{(t)}(\tilde{X},X)} \cdot \big(f^{(t)}(X) - Y\big).$$

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Neural Tangent Kernel Theory

Consider *squared loss* $\mathcal{L}(\hat{Y}, Y) = \frac{1}{2N}\|(\hat{Y} - Y)\|_2^2$ and for simplicity set $M_L = 1$. Then the gradient flow dynamics of a DNN takes form:

$$\dot{\theta}^{(t)} = -\nabla_\theta \mathcal{L}(f^{(t)}(X), Y) = -\frac{1}{N}\nabla_\theta f^{(t)}(X)^T \cdot \big(f^{(t)}(X) - Y\big),$$

$$\dot{f}^{(t)}(\tilde{X}) = \nabla_\theta f^{(t)}(\tilde{X}) \cdot \dot{\theta}^{(t)} = -\frac{1}{N}\underbrace{\nabla_\theta f^{(t)}(\tilde{X})\nabla_\theta f^{(t)}(X)^T}_{\Theta^{(t)}(\tilde{X},X)} \cdot \big(f^{(t)}(X) - Y\big).$$

Definition: *Neural tangent kernel (NTK)* of a DNN with output function f and trainable parameters θ is given by

$$\Theta(x_i, x_j) := \nabla_\theta f(x_i)^T \nabla_\theta f(x_j), \quad x_i, x_j \in \mathbb{R}^{M_0}.$$

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Neural Tangent Kernel Theory

Results on infinite-width limit of NTK $M_\ell \to \infty, \ell = 1, \ldots, L-1$**:**[1]

LUDWIG-
MAXIMILIANS-
UNIVERSITÄT
MÜNCHEN

Neural Tangent Kernel Theory

Results on infinite-width limit of NTK $M_\ell \to \infty, \ell = 1, \dots, L-1$:[1]

- NTK is *deterministic under random initialization*:

$$\Theta^{(0)}(x_i, x_j) \to \mathbb{E}_\theta[\Theta^{(0)}(x_i, x_j)] = \Theta^*(x_i, x_j),$$

$$\text{where } \mathrm{W}_{ij}^\ell = \frac{\sigma_w}{\sqrt{M^\ell}} \mathrm{w}_{ij}^\ell, \quad \mathrm{w}_{ij}^\ell \sim \mathcal{N}(0,1),$$

$$\mathrm{b}_i^\ell = \sigma_b \beta_i^\ell, \quad \beta_i^\ell \sim \mathcal{N}(0,1).$$

LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Neural Tangent Kernel Theory

Results on infinite-width limit of NTK $M_\ell \to \infty, \ell = 1, \ldots, L-1$**:**[1]

- NTK is *deterministic under random initialization*:

$$\Theta^{(0)}(x_i, x_j) \to \mathbb{E}_\theta[\Theta^{(0)}(x_i, x_j)] = \Theta^*(x_i, x_j),$$

$$\text{where } \mathrm{W}_{ij}^\ell = \frac{\sigma_w}{\sqrt{M^\ell}} \mathrm{w}_{ij}^\ell, \quad \mathrm{w}_{ij}^\ell \sim \mathcal{N}(0,1),$$
$$\mathrm{b}_i^\ell = \sigma_b \beta_i^\ell, \quad \beta_i^\ell \sim \mathcal{N}(0,1).$$

- NTK stays *constant during training*:

$$\Theta^{(t)}(x_i, x_j) \to \Theta^*(x_i, x_j).$$

Neural Tangent Kernel Theory

Results on infinite-width limit of NTK $M_\ell \to \infty, \ell = 1, \dots, L-1$**:**[1]

- NTK is *deterministic under random initialization*:

$$\Theta^{(0)}(x_i, x_j) \to \mathbb{E}_\theta[\Theta^{(0)}(x_i, x_j)] = \Theta^*(x_i, x_j),$$

$$\text{where } \mathrm{W}_{ij}^\ell = \frac{\sigma_w}{\sqrt{M^\ell}} \mathrm{w}_{ij}^\ell, \quad \mathrm{w}_{ij}^\ell \sim \mathcal{N}(0,1),$$
$$\mathrm{b}_i^\ell = \sigma_b \beta_i^\ell, \quad \beta_i^\ell \sim \mathcal{N}(0,1).$$

- NTK stays *constant during training*:

$$\Theta^{(t)}(x_i, x_j) \to \Theta^*(x_i, x_j).$$

Then in the infinite-width limit gradient flow dynamics with squared loss:

$$\dot{f}^{(t)}(\tilde{X}) = -\frac{1}{N} \Theta^*(\tilde{X}, X) \cdot \left(f^{(t)}(X) - Y\right)$$

Neural Tangent Kernel Theory

Results on infinite-width limit of NTK $M_\ell \to \infty, \ell = 1, \ldots, L-1$**:**[1]

- NTK is *deterministic under random initialization*:

$$\Theta^{(0)}(x_i, x_j) \to \mathbb{E}_\theta[\Theta^{(0)}(x_i, x_j)] = \Theta^*(x_i, x_j),$$

$$\text{where } \mathrm{W}_{ij}^\ell = \frac{\sigma_w}{\sqrt{M^\ell}} \mathrm{w}_{ij}^\ell, \quad \mathrm{w}_{ij}^\ell \sim \mathcal{N}(0,1),$$

$$\mathrm{b}_i^\ell = \sigma_b \beta_i^\ell, \quad \beta_i^\ell \sim \mathcal{N}(0,1).$$

- NTK stays *constant during training*:

$$\Theta^{(t)}(x_i, x_j) \to \Theta^*(x_i, x_j).$$

Then in the infinite-width limit gradient flow dynamics with squared loss:

$$\dot{f}^{(t)}(\tilde{X}) = -\frac{1}{N} \Theta^*(\tilde{X}, X) \cdot \left(f^{(t)}(X) - Y\right)$$

$\rightsquigarrow$ *Infinitely-wide DNNs evolve as kernel regression with NTK kernel!*

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Do finite DNNs behave as infinite-width ones?

Problems:

- If NTK matrix is constant, *no feature learning occurs*.

Do finite DNNs behave as infinite-width ones?

Problems:

- If NTK matrix is constant, *no feature learning occurs*.
- Empirical performance of NTK and finite DNNs differs.[2],[3]

Do finite DNNs behave as infinite-width ones?

Problems:

- If NTK matrix is constant, *no feature learning occurs*.
- Empirical performance of NTK and finite DNNs differs.[2],[3]
- In infinite-*depth-and-width* limit ($L/M > 0$), NTK of ReLU DNNs initialized with $\sigma_w^2 = 2, \sigma_b^2 = 0$ is random.[4]

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Do finite DNNs behave as infinite-width ones?

Problems:

- If NTK matrix is constant, *no feature learning occurs*.
- Empirical performance of NTK and finite DNNs differs.[2],[3]
- In infinite-*depth-and-width* limit ($L/M > 0$), NTK of ReLU DNNs initialized with $\sigma_w^2 = 2, \sigma_b^2 = 0$ is random.[4]

$\rightsquigarrow$ *It is not clear when NTK theory explains DNNs' behavior!*

Do finite DNNs behave as infinite-width ones?

Problems:

- If NTK matrix is constant, *no feature learning occurs*.
- Empirical performance of NTK and finite DNNs differs.[2],[3]
- In infinite-*depth-and-width* limit ($L/M > 0$), NTK of ReLU DNNs initialized with $\sigma_w^2 = 2, \sigma_b^2 = 0$ is random.[4]

$\rightsquigarrow$ *It is not clear when NTK theory explains DNNs' behavior!*

Our contributions:

- Study *ReLU* and *sigmoid* DNNs with various hyperparameters $(\sigma_w, \sigma_b, L, M)$.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Do finite DNNs behave as infinite-width ones?

Problems:

- If NTK matrix is constant, *no feature learning occurs*.
- Empirical performance of NTK and finite DNNs differs.[2],[3]
- In infinite-*depth-and-width* limit ($L/M > 0$), NTK of ReLU DNNs initialized with $\sigma_w^2 = 2, \sigma_b^2 = 0$ is random.[4]

$\rightsquigarrow$ *It is not clear when NTK theory explains DNNs' behavior!*

Our contributions:

- Study *ReLU* and *sigmoid* DNNs with various hyperparameters $(\sigma_w, \sigma_b, L, M)$.
- Identify *two phases* in hyperparameter space where NTK regime does and does not hold.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Do finite DNNs behave as infinite-width ones?

Problems:

- If NTK matrix is constant, *no feature learning occurs*.
- Empirical performance of NTK and finite DNNs differs.[2],[3]
- In infinite-*depth-and-width* limit ($L/M > 0$), NTK of ReLU DNNs initialized with $\sigma_w^2 = 2, \sigma_b^2 = 0$ is random.[4]

$\rightsquigarrow$ *It is not clear when NTK theory explains DNNs' behavior!*

Our contributions:

- Study *ReLU* and *sigmoid* DNNs with various hyperparameters $(\sigma_w, \sigma_b, L, M)$.
- Identify *two phases* in hyperparameter space where NTK regime does and does not hold.
- Study variance of DNNs output $Var_{\theta,D}\left[f^{(t\rightarrow\infty)}(x)\right]$ under NTK theory.

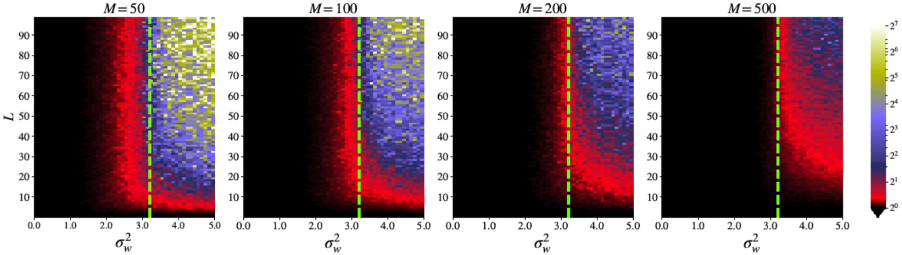
LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Randomness at initialization

Setup:

- Fully-connected *tanh* networks with L layers and constant width M.
- Initialized as $W_{ij}^{\ell} \sim \mathcal{N}\left(0, \frac{\sigma_w^2}{M}\right), b_i^{\ell} \sim \mathcal{N}(0, \sigma_b^2)$

$\frac{\mathbb{E}_\theta[\Theta^{(0)}(x,x)^2]}{\mathbb{E}_\theta^2[\Theta^{(0)}(x,x)]}$ ratio measures randomness at initialization:



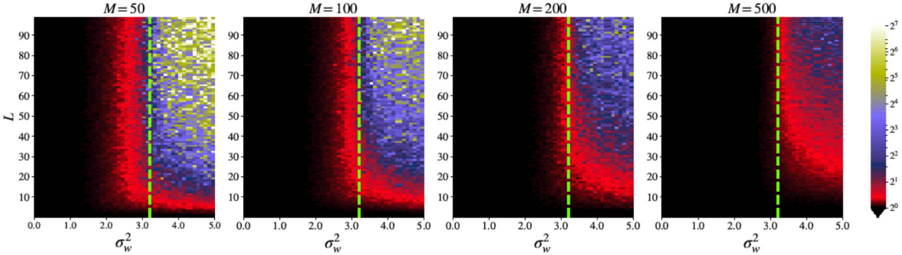
We use standard parametrization instead of NTK parametrization here. However, for constant-width networks this does not affect the results.

Randomness at initialization

Setup:

- Fully-connected *tanh* networks with L layers and constant width M.
- Initialized as $W^{\ell}_{ij} \sim \mathcal{N}\left(0, \frac{\sigma_w^2}{M}\right), b^{\ell}_i \sim \mathcal{N}(0, \sigma_b^2)$

$$\frac{\mathbb{E}_\theta[\Theta^{(0)}(x,x)^2]}{\mathbb{E}^2_\theta[\Theta^{(0)}(x,x)]} \text{ ratio measures randomness at initialization:}$$



$\rightsquigarrow$ *Deep NNs with large σ_w are random at initialization!*

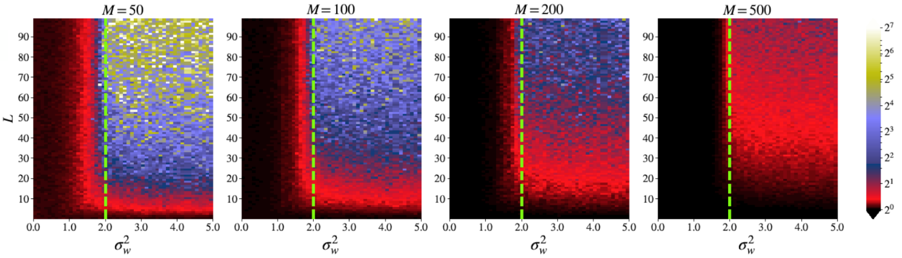
We use standard parametrization instead of NTK parametrization here. However, for constant-width networks this does not affect the results.

Randomness at initialization

Setup:

- Fully-connected *ReLU* networks with L layers and constant width M.
- Initialized as $W_{ij}^{\ell} \sim \mathcal{N}\left(0, \frac{\sigma_w^2}{M}\right), b_i^{\ell} \sim \mathcal{N}(0, \sigma_b^2)$

$\frac{\mathbb{E}_\theta[\Theta^{(0)}(x,x)^2]}{\mathbb{E}_\theta^2[\Theta^{(0)}(x,x)]}$ ratio measures randomness at initialization:



$\rightsquigarrow$ *Deep NNs with large σ_w are random at initialization!*

We use standard parametrization instead of NTK parametrization here. However, for constant-width networks this does not affect the results.

Vanishing and exploding gradients

Behaviour of *gradients at initialization* is controlled by variable χ:[5]

$$\chi := \sigma_w^2 \int \left[\phi'\left(\sqrt{q^*}v\right)\right]^2 Dv, \quad Dv = \frac{dv}{\sqrt{2\pi}} e^{-v^2/2},$$

where $q^* = \lim_{\ell\to\infty} q^\ell$ and $q^\ell := \frac{1}{M_\ell}\sum_{k=1}^{M_\ell}\left(z_k^\ell\right)^2$ is the pre-activation "length" in layer ℓ.

$q^\ell(x)$ depends only on the norm of x. Therefore, for simplicity of notation we can assume normalized inputs and omit argument x here.

Vanishing and exploding gradients

Behaviour of *gradients at initialization* is controlled by variable χ:[5]

$$\chi := \sigma_w^2 \int \left[\phi'\left(\sqrt{q^*}v\right)\right]^2 Dv, \quad Dv = \frac{dv}{\sqrt{2\pi}} e^{-v^2/2},$$

where $q^* = \lim_{\ell\to\infty} q^\ell$ and $q^\ell := \frac{1}{M_\ell}\sum_{k=1}^{M_\ell}\left(z_k^\ell\right)^2$ is the pre-activation "length" in layer ℓ.

$\rightsquigarrow$ *χ depends on hyperparameters (σ_w, σ_b) and activation function ϕ.*

$q^\ell(x)$ depends only on the norm of x. Therefore, for simplicity of notation we can assume normalized inputs and omit argument x here.

Vanishing and exploding gradients

Behaviour of *gradients at initialization* is controlled by variable χ:[5]

$$\chi := \sigma_w^2 \int \left[\phi'\left(\sqrt{q^*}v\right)\right]^2 Dv, \quad Dv = \frac{dv}{\sqrt{2\pi}} e^{-v^2/2},$$

where $q^* = \lim_{\ell \to \infty} q^\ell$ and $q^\ell := \frac{1}{M_\ell} \sum_{k=1}^{M_\ell} \left(z_k^\ell\right)^2$ is the pre-activation "length" in layer ℓ.

$\rightsquigarrow$ *χ depends on hyperparameters (σ_w, σ_b) and activation function ϕ.*

We can identify the following situations based on χ:

- *Chaotic phase:* If $\chi > 1$, gradients explode as they backpropagate.

$q^\ell(x)$ depends only on the norm of x. Therefore, for simplicity of notation we can assume normalized inputs and omit argument x here.

Vanishing and exploding gradients

Behaviour of *gradients at initialization* is controlled by variable χ:[5]

$$\chi := \sigma_w^2 \int \left[\phi'\left(\sqrt{q^*}v\right)\right]^2 Dv, \quad Dv = \frac{dv}{\sqrt{2\pi}} e^{-v^2/2},$$

where $q^* = \lim_{\ell\to\infty} q^\ell$ and $q^\ell := \frac{1}{M_\ell}\sum_{k=1}^{M_\ell}\left(z_k^\ell\right)^2$ is the pre-activation "length" in layer ℓ.

$\rightsquigarrow$ *χ depends on hyperparameters (σ_w, σ_b) and activation function ϕ.*

We can identify the following situations based on χ:

- *Chaotic phase:* If $\chi > 1$, gradients explode as they backpropagate.
- *Ordered phase:* If $\chi < 1$, gradients vanish.

$q^\ell(x)$ depends only on the norm of x. Therefore, for simplicity of notation we can assume normalized inputs and omit argument x here.

Vanishing and exploding gradients

Behaviour of *gradients at initialization* is controlled by variable χ:[5]

$$\chi := \sigma_w^2 \int \left[\phi'\left(\sqrt{q^*}v\right)\right]^2 Dv, \quad Dv = \frac{dv}{\sqrt{2\pi}} e^{-v^2/2},$$

where $q^* = \lim_{\ell\to\infty} q^\ell$ and $q^\ell := \frac{1}{M_\ell}\sum_{k=1}^{M_\ell}\left(z_k^\ell\right)^2$ is the pre-activation "length" in layer ℓ.

$\rightsquigarrow$ *χ depends on hyperparameters (σ_w, σ_b) and activation function ϕ.*

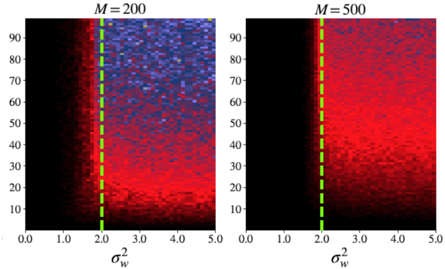
We can identify the following situations based on χ:

- *Chaotic phase:* If $\chi > 1$, gradients explode as they backpropagate.
- *Ordered phase:* If $\chi < 1$, gradients vanish.
- *≪Edge of chaos≫ (EOC):* $\chi \approx 1$ allows deeper signal propagation.

$q^\ell(x)$ depends only on the norm of x. Therefore, for simplicity of notation we can assume normalized inputs and omit argument x here.

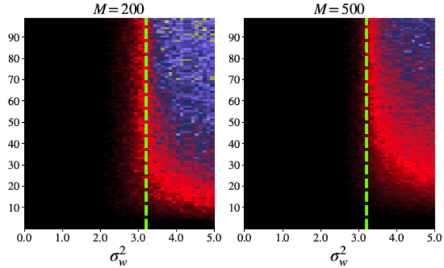
Randomness at initialization

ReLU DNNs



$\chi = 1$ if $\sigma_w^2 = 2$

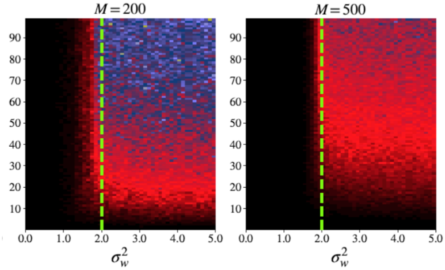
tanh DNNs



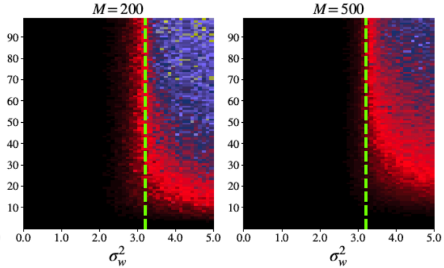
$\chi = 1$ if $\sigma_w^2 \approx 3.2$

Randomness at initialization

ReLU DNNs



tanh DNNs



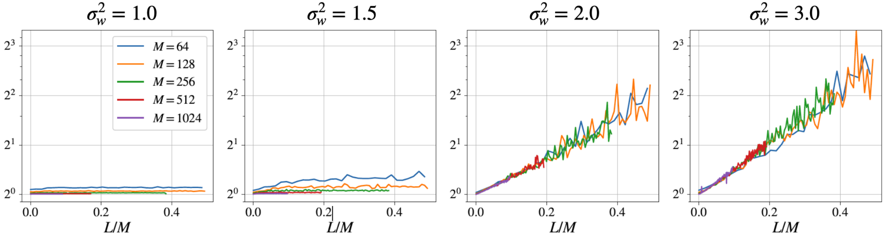
$\chi = 1$ if $\sigma_w^2 = 2$

$\chi = 1$ if $\sigma_w^2 \approx 3.2$

$\rightsquigarrow$ *Deep networks in chaotic phase are random at initialization!*

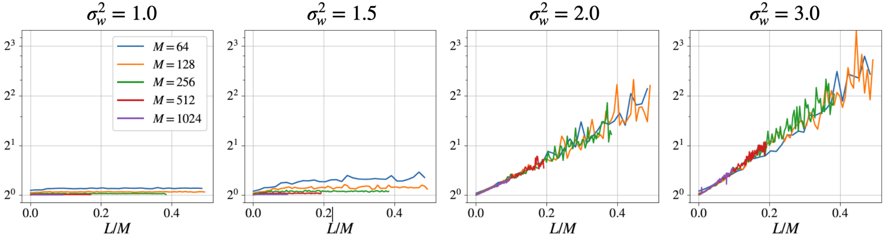
Randomness at initialization

$\frac{\mathbb{E}_\theta[\Theta^{(0)}(x,x)^2]}{\mathbb{E}^2_\theta[\Theta^{(0)}(x,x)]}$ ratio as a function of $\frac{L}{M}$:



Randomness at initialization

$\frac{\mathbb{E}_\theta[\Theta^{(0)}(x,x)^2]}{\mathbb{E}^2_\theta[\Theta^{(0)}(x,x)]}$ ratio as a function of $\frac{L}{M}$:



$\rightsquigarrow$ *Exponential growth in L/M in the chaotic phase.*

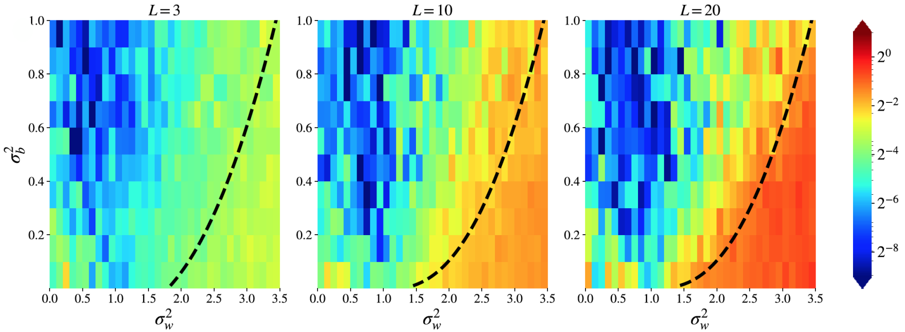
$\rightsquigarrow$ *Dependence on $1/M$ in the ordered phase.*

Change during training

Setup:

- Fully-connected *tanh* networks with L layers and constant width $M = 256$.

  $\frac{\|\Theta^{(t)} - \Theta^{(0)}\|_F}{\|\Theta^{(0)}\|_F}$ shows if NTK changes significantly during training:



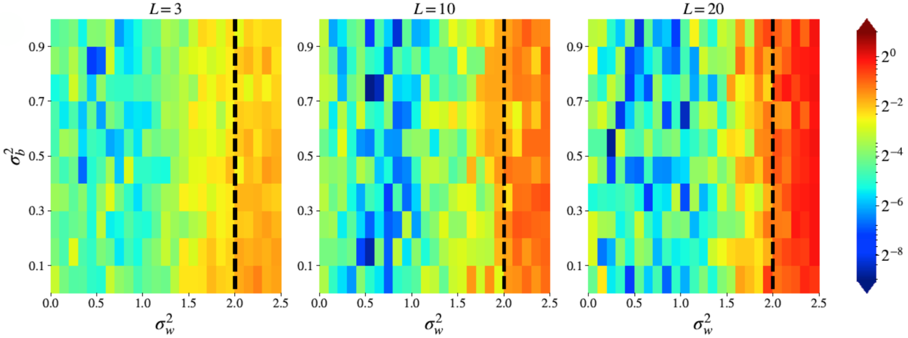
LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Change during training

Setup:

- Fully-connected *ReLU* networks with L layers and constant width $M = 256$.

$\frac{\|\Theta^{(t)} - \Theta^{(0)}\|_F}{\|\Theta^{(0)}\|_F}$ shows if NTK changes significantly during training:

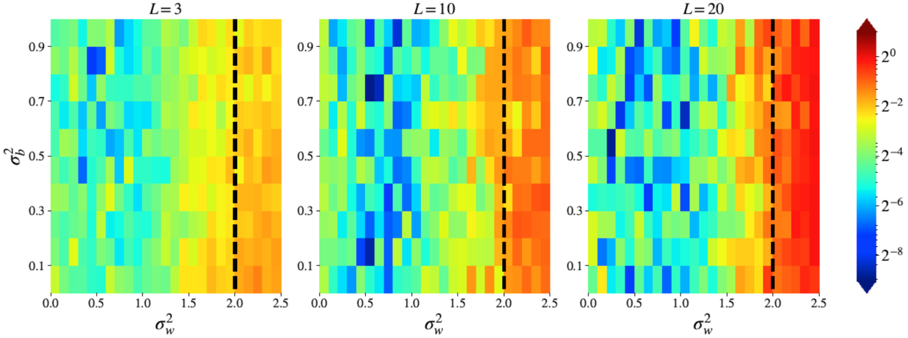


Change during training

Setup:

- Fully-connected *ReLU* networks with L layers and constant width $M = 256$.

$\frac{\|\Theta^{(t)} - \Theta^{(0)}\|_F}{\|\Theta^{(0)}\|_F}$ shows if NTK changes significantly during training:



$\rightsquigarrow$ *NTK changes significantly during training in the chaotic phase.*

Generalization in the NTK regime

NTK has the following *structure* at initialization:

$$\Theta^*(X) = \bar{\Theta}^*(\mathbb{I}_N + \epsilon(X)),$$
$$\bar{\Theta}^* = (\bar{\kappa}_1 - \bar{\kappa}_2)\mathbb{I}_N + \bar{\kappa}_2 \mathbb{1}_N \mathbb{1}_N^T,$$

where $\epsilon(X) \xrightarrow[L\to\infty]{} 0$ [6] is the only data-dependent part and $\bar{\kappa}_i, i = 1, 2$ are controlled by depth and gradients' behaviour.

Generalization in the NTK regime

NTK has the following *structure* at initialization:

$$\Theta^*(X) = \bar{\Theta}^*(\mathbb{I}_N + \epsilon(X)),$$
$$\bar{\Theta}^* = (\bar{\kappa}_1 - \bar{\kappa}_2)\mathbb{I}_N + \bar{\kappa}_2 \mathbb{1}_N \mathbb{1}_N^T,$$

where $\epsilon(X) \xrightarrow[L\to\infty]{} 0$ [6] is the only data-dependent part and $\bar{\kappa}_i, i = 1, 2$ are controlled by depth and gradients' behaviour.

NTK behaviour depends on initialization:

- *Chaotic phase:* $\bar{\kappa}_1/\bar{\kappa}_2 \gg 1$ for large $L \Rightarrow \Theta^* \approx \bar{\kappa}_1 \mathbb{I}_N$.
- *Ordered phase:* $\bar{\kappa}_1/\bar{\kappa}_2 \approx 1$ for large $L \Rightarrow \Theta^* \approx \bar{\kappa}_2 \mathbb{1}_N \mathbb{1}_N^T$.

Generalization in the NTK regime

NTK has the following *structure* at initialization:

$$\Theta^*(X) = \bar{\Theta}^*(\mathbb{I}_N + \epsilon(X)),$$
$$\bar{\Theta}^* = (\bar{\kappa}_1 - \bar{\kappa}_2)\mathbb{I}_N + \bar{\kappa}_2 \mathbb{1}_N \mathbb{1}_N^T,$$

where $\epsilon(X) \xrightarrow[L\to\infty]{} 0$ [6] is the only data-dependent part and $\bar{\kappa}_i, i = 1, 2$ are controlled by depth and gradients' behaviour.

NTK behaviour depends on initialization:

- *Chaotic phase:* $\bar{\kappa}_1/\bar{\kappa}_2 \gg 1$ for large $L \Rightarrow \Theta^* \approx \bar{\kappa}_1 \mathbb{I}_N$.
- *Ordered phase:* $\bar{\kappa}_1/\bar{\kappa}_2 \approx 1$ for large $L \Rightarrow \Theta^* \approx \bar{\kappa}_2 \mathbb{1}_N \mathbb{1}_N^T$.

$\rightsquigarrow$ *DNNs in the NTK regime have different dynamics in ordered and chaotic phases!*

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Generalization in the NTK regime

Theorem (Seleznova&Kutyniok, 2020): Assume the NTK matrix is well-conditioned ($\bar{\kappa}_1/\bar{\kappa}_2 \gg 1$). Then for the *variance of a trained DNN in the NTK regime* we have:

$$Var_{\theta,X}[f^{(t\to\infty)}(\tilde{x})] \approx (1+\frac{A^2}{N})\Big(Var^{(0)} - Cov^{(0)}\Big) \\ +(A-1)^2 Cov^{(0)},$$

where $A = \frac{N}{\bar{\kappa}_1/\bar{\kappa}_2+(N-1)}$, $Var^{(0)} := Var_{\theta,X,\tilde{x}}[f^{(0)}(\tilde{x})]$ is the output variance at initialization, $Cov^{(0)} = Cov_{\theta,X,x_i\neq x_j}[f^{(0)}(x_i), f^{(0)}(x_j)]$ is the output covariance on two different inputs.

Generalization in the NTK regime

Theorem (Seleznova&Kutyniok, 2020): Assume the NTK matrix is well-conditioned ($\bar{\kappa}_1/\bar{\kappa}_2 \gg 1$). Then for the *variance of a trained DNN in the NTK regime* we have:

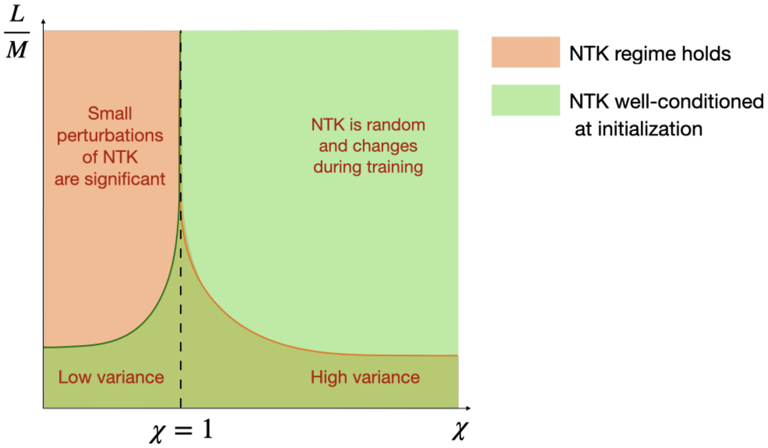
$$Var_{\theta,X}[f^{(t\to\infty)}(\tilde{x})] \approx (1+\frac{A^2}{N})\Big(Var^{(0)} - Cov^{(0)}\Big) + (A-1)^2 Cov^{(0)},$$

where $A = \frac{N}{\bar{\kappa}_1/\bar{\kappa}_2 + (N-1)}$, $Var^{(0)} := Var_{\theta,X,\tilde{x}}[f^{(0)}(\tilde{x})]$ is the output variance at initialization, $Cov^{(0)} = Cov_{\theta,X,x_i\neq x_j}[f^{(0)}(x_i), f^{(0)}(x_j)]$ is the output covariance on two different inputs.

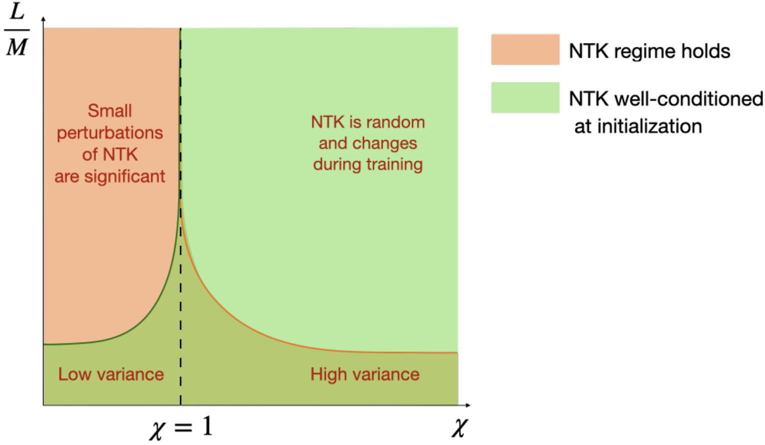
If all the conditions hold, we have:

- *Chaotic phase:* $Var_{\theta,X}[f^{(t\to\infty)}(\tilde{x})] \propto Var^{(0)}$ – large variance, which growth with depth L.
- *Ordered phase:* $Var_{\theta,X}[f^{(t\to\infty)}(\tilde{x})] \approx 0$ – low variance for large L.

When can we trust the results?



When can we trust the results?



$\rightsquigarrow$ *Deep networks cannot be analyzed within the NTK theory!*

Conclusions

- NTK theory is a powerful tool to analyze DNNs theoretically. However, it is important to understand when it is applicable.

Conclusions

- NTK theory is a powerful tool to analyze DNNs theoretically. However, it is important to understand when it is applicable.
- *Empirical NTK* behaves as theoretical NTK for DNNs in the ordered phase but not in the chaotic phase.

Conclusions

- NTK theory is a powerful tool to analyze DNNs theoretically. However, it is important to understand when it is applicable.
- *Empirical NTK* behaves as theoretical NTK for DNNs in the ordered phase but not in the chaotic phase.
- Generalization of *shallow networks* ($L/M \approx 0$) can be analyzed within the NTK theory.

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Conclusions

- NTK theory is a powerful tool to analyze DNNs theoretically. However, it is important to understand when it is applicable.
- *Empirical NTK* behaves as theoretical NTK for DNNs in the ordered phase but not in the chaotic phase.
- Generalization of *shallow networks* ($L/M \approx 0$) can be analyzed within the NTK theory.
- *Deep networks* are hard to analyze within the NTK theory.
  $\rightsquigarrow$ New approaches are needed to analyze DNNs theoretically.

References:

[1] Jacot et al. *Neural Tangent Kernel: Convergence and Generalization in Neural Networks.* 2018
[2] Lee et al. *Finite Versus Infinite Neural Networks: an Empirical Study.* 2020
[3] Bai & Lee. *Beyond Linearization: On Quadratic and Higher-Order Approximation of Wide Neural Networks.* 2020
[4] Hanin & Nica. *Finite Depth and Width Corrections to the Neural Tangent Kernel.* 2020
[5] Schoenholz et al. *Deep information propagation.* 2017
[6] Xiao et al. *Disentangling Trainability and Generalization in Deep Neural Networks.* 2020

Thank you for your attention!

Parametrization*

Infinite-width limit of NTK is normally considered in *NTK parametrization (NTP)* instead of *standard parametrization (SP)*.

$$\text{SP:} \quad a^{l+1} = \phi\Big(\mathrm{W}^l a^l + \mathrm{b}^l\Big) \qquad \text{NTP:} \quad a^{l+1} = \phi\Big(\frac{\sigma_w}{\sqrt{M^l}} \mathrm{w}^l x^l + \sigma_b \mathrm{b}^l\Big)$$

$$\mathrm{W}^l_{ij} \sim \mathcal{N}\big(0, \frac{\sigma_w^2}{M^l}\big), \mathrm{b}^l_i \sim \mathcal{N}(0, \sigma_b^2) \qquad \mathrm{w}^l_{ij} \sim \mathcal{N}\big(0, 1\big), \mathrm{b}^l_i \sim \mathcal{N}(0, 1)$$

The change from SP to NTK amounts to: $\nabla_{W^l} f^{(t)}(x) \to \frac{1}{\sqrt{M^l}} \nabla_{W^l} f^{(t)}(x)$
And for constant-width networks: $\Theta^{(t)}(x_i, x_j) \rightarrow \frac{1}{M} \Theta^{(t)}(x_i, x_j)$

$\rightsquigarrow$ *The same dynamics of $f^{(t)}$ with proper adjustment of η.*

$\rightsquigarrow$ $\frac{\mathbb{E}[\Theta^{(0)}(x,x)^2]}{\mathbb{E}^2[\Theta^{(0)}(x,x)]}$ *and* $\frac{\|\Theta^{(t)} - \Theta^{(0)}\|_F}{\|\Theta^{(0)}\|_F}$ *ratios are not affected.*