The Role of Over-parameterization in Statistical Machine Learning

- a function space perspective

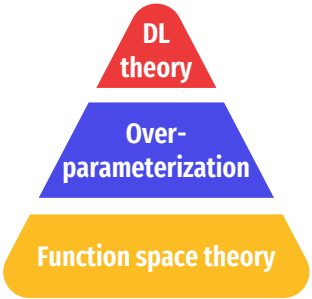
Fanghui Liu

Laboratory for Information and Inference Systems (LIONS)
École Polytechnique Fédérale de Lausanne (EPFL)
Switzerland

at Deep Learning and Optimization Seminar

lions@epfl EPFL

Today's Over-parameterization journey - *the good, the bad, the ugly*

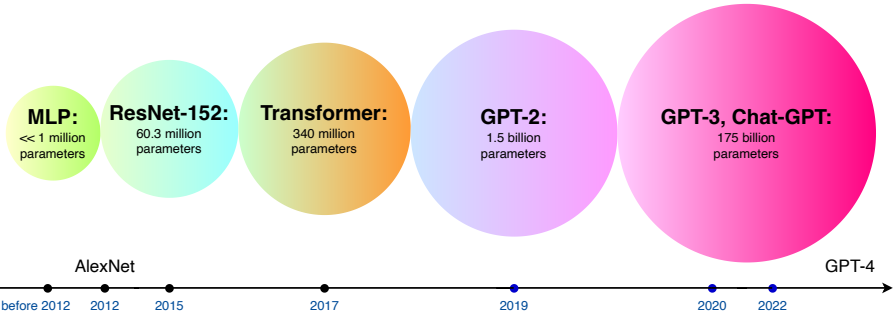


Helps robustness or not?

Why generalize well?

Motivation

Over-parameterization: more parameters than training data



Over-parameterization: more parameters than training data

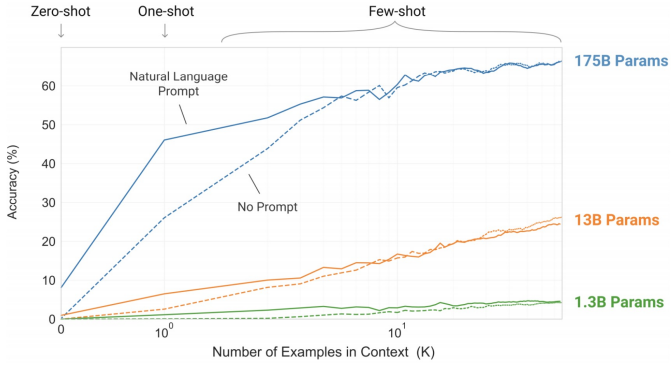


Figure: Larger models make increasingly efficient use of in-context information: source from Open AI.

DNNs: the good in fitting ...

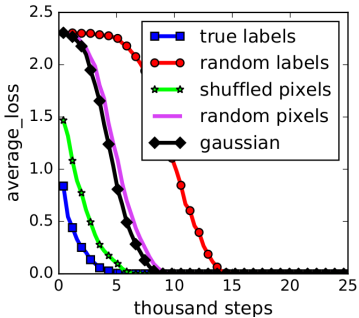


Figure: DNN Training curves on CIFAR10, from [1]

- A gap between theory and practice:
  - DNNs can fit random labels
  - SGD: zero training error and low test error

DNNs: the bad in robustness...

(a) Invisibility [2]

(b) Stop sign classified as 45 mph sign [3]

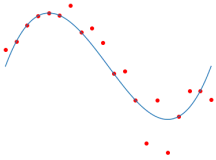
DNNs: the bad in robustness...

(a) Invisibility [2]

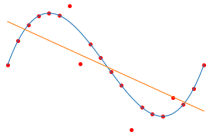
(b) Stop sign classified as 45 mph sign [3]

the ugly in over-parameterization?

A toy example: curve fitting

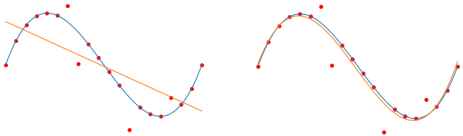


A toy example: curve fitting



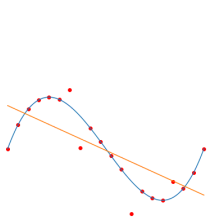
(a) under-fitting

A toy example: curve fitting

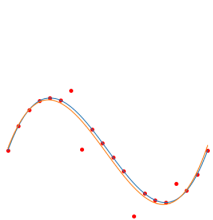


(a) under-fitting (b) sweet spot

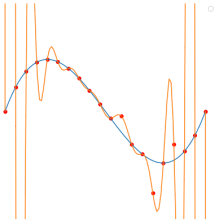
A toy example: curve fitting



(a) under-fitting

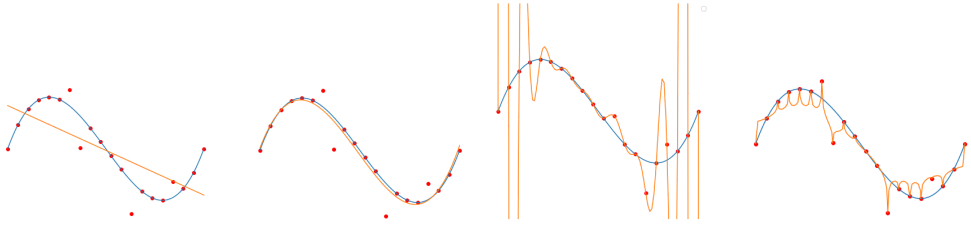


(b) sweet spot



(c) overfitting

A toy example: curve fitting



(a) under-fitting (b) sweet spot (c) overfitting (d) benign overfitting [4]

Figure: Test performance on curve fitting: source from Open AI.

Benign overfitting and double descent

benign overfitting [4, 5, 6]:

- model is very complex
- perfectly fit noisy data and generalize well

Benign overfitting and double descent

benign overfitting [4, 5, 6]:

- model is very complex
- perfectly fit noisy data and generalize well

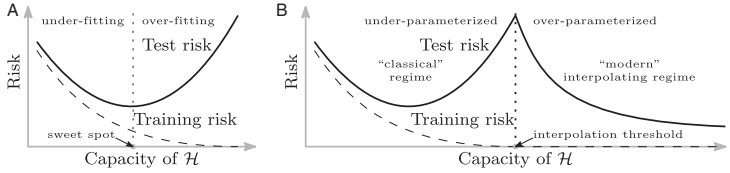
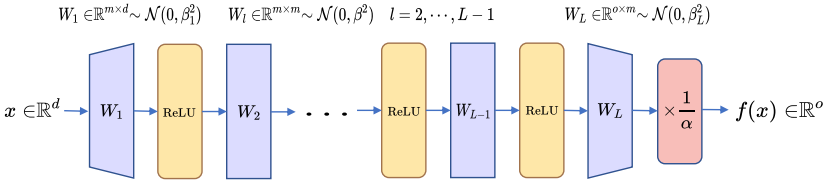


Figure: classical learning theory vs. double descent: source from [7].

Typical architecture of DNNs



Initialization	Formulation
LeCun initialization	$\beta_1 = \sqrt{\frac{1}{d}}$, $\beta = \beta_L = \sqrt{\frac{1}{m}}$
He initialization	$\beta_1 = \sqrt{\frac{2}{d}}$, $\beta = \beta_L = \sqrt{\frac{2}{m}}$
NTK initialization	$\beta = \beta_1 = \sqrt{\frac{2}{m}}$, $\beta_L = 1$
Xavier initialization	$\beta_1 = \frac{2}{m+d}$, $\beta = \frac{1}{m}$, $\beta_L = \frac{2}{m+1}$

Why function space theory is needed? (lazy training regime)

$$\mathcal{F}_{\mathrm{NN},m} = \left\{ f_m(\boldsymbol{x};\boldsymbol{\Theta}) = \sum_{i=1}^{m} a_i \max\left(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle, 0\right) : a_i \in \mathbb{R}, \boldsymbol{w}_i \in \mathbb{R}^d \right\}$$

Why function space theory is needed? (lazy training regime)

$$\mathcal{F}_{\mathrm{NN},m} = \left\{ f_m(\boldsymbol{x};\boldsymbol{\Theta}) = \sum_{i=1}^{m} a_i \max\left(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle, 0\right) : a_i \in \mathbb{R}, \boldsymbol{w}_i \in \mathbb{R}^d \right\}$$

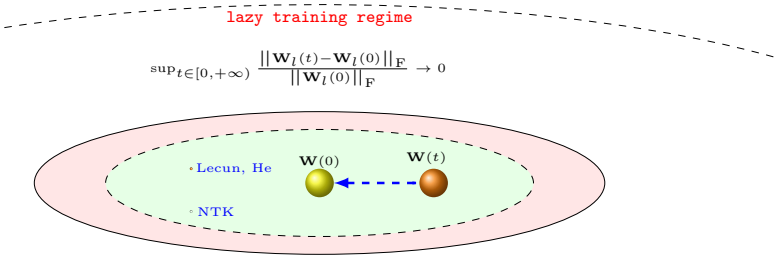
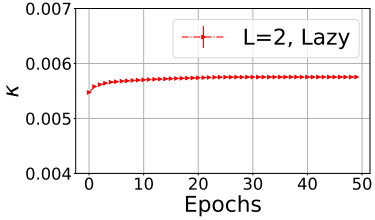


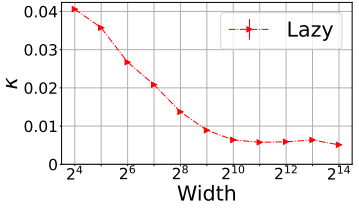
Figure: Training dynamics of two-layer ReLU NNs under different initializations [8, 9, 10].

Why function space theory is needed? (lazy training regime)

$$\mathcal{F}_{\mathrm{NN},m} = \left\{ f_m(\boldsymbol{x};\boldsymbol{\Theta}) = \sum_{i=1}^{m} a_i \max\left(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle, 0\right) : a_i \in \mathbb{R}, \boldsymbol{w}_i \in \mathbb{R}^d \right\}$$

$$\text{lazy training ratio } \kappa := \frac{\sum_{l=1}^{L} \|\boldsymbol{W}_l(t) - \boldsymbol{W}_l(0)\|_{\mathrm{F}}}{\sum_{l=1}^{L} \|\boldsymbol{W}_l(0)\|_{\mathrm{F}}}$$





Why function space theory is needed? (non-lazy training regime)

$$\mathcal{F}_{\mathrm{NN},m} = \left\{ f_m(\boldsymbol{x};\boldsymbol{\Theta}) = \sum_{i=1}^{m} a_i \max\left(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle, 0\right) : \frac{1}{m}\sum_{i=1}^{m} |a_i| \|\boldsymbol{w}_i\|_1 < \infty \right\} \quad \text{(Barron space)}$$

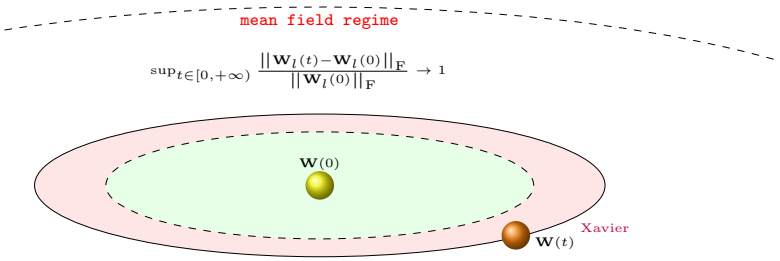


Figure: Training dynamics of two-layer ReLU NNs under different initializations [8, 9, 10].

Why function space theory is needed? (non-lazy training regime)

$$\mathcal{F}_{\mathrm{NN},m} = \left\{ f_m(\boldsymbol{x};\boldsymbol{\Theta}) = \sum_{i=1}^{m} a_i \max\left(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle, 0\right) : \frac{1}{m}\sum_{i=1}^{m} |a_i| \|\boldsymbol{w}_i\|_1 < \infty \right\} \quad \text{(Barron space)}$$

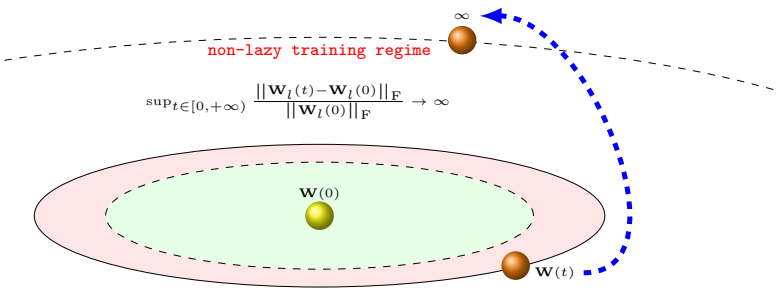
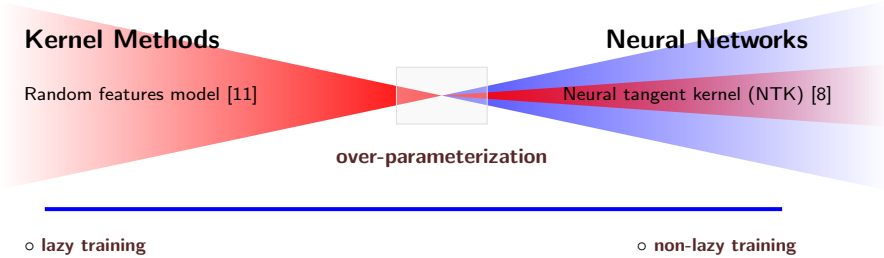


Figure: Training dynamics of two-layer ReLU NNs under different initializations [8, 9, 10].

Research Overview: Today's talk



- **generalization** of NNs
  - [LSC, NeurIPS22] why **over-parameterized NNs** generalize well under SGD training?
- **robustness** of NNs
  - [ZLCC, NeurIPS22] **Over-parameterization** helps or hurts robustness of NNs?

Obstacle in modern ML setting

Modern ML setting

comparably large: #training data n, #parameters m, input dimension d

- high dimensional setting [12, 13]: $c \leqslant \{d/n, m/n\} \leqslant C$

Obstacle in modern ML setting

Modern ML setting

comparably large: #training data n, #parameters m, input dimension d

- high dimensional setting [12, 13]: $c \leqslant \{d/n, m/n\} \leqslant C$

$$\text{ridge regression: } \hat{\boldsymbol{\beta}} = (\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I})^{-1} \boldsymbol{X}^\top \boldsymbol{y} \xrightarrow{?} \boldsymbol{\beta}^\star := \arg\min_{\boldsymbol{\beta}} \mathbb{E}[(\boldsymbol{y} - \boldsymbol{\beta}^\top \boldsymbol{x})^2]$$

Obstacle in modern ML setting

Modern ML setting

comparably large: #training data n, #parameters m, input dimension d

- high dimensional setting [12, 13]: $c \leqslant \{d/n, m/n\} \leqslant C$

$$\text{ridge regression: } \hat{\boldsymbol{\beta}} = (\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I})^{-1} \boldsymbol{X}^\top \boldsymbol{y} \xrightarrow{?} \boldsymbol{\beta}^\star := \arg\min_{\boldsymbol{\beta}} \mathbb{E}[(\boldsymbol{y} - \boldsymbol{\beta}^\top \boldsymbol{x})^2]$$

Consistency of estimator?

- classical low-dimensional setting (d fixed and $n \to \infty$): ✓
- classical high-dimensional setting ($d \gg n$) if sparsity [14]: ✓
- n, d, m are comparably large: ✗

Obstacle in modern ML setting

Modern ML setting

comparably large: #training data n, #parameters m, input dimension d

- high dimensional setting [12, 13]: $c \leqslant \{d/n, m/n\} \leqslant C$

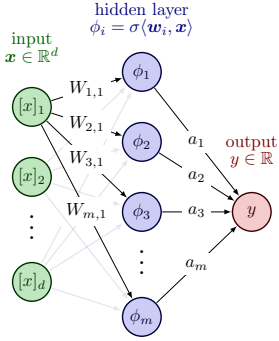
ridge regression: $\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I})^{-1} \boldsymbol{X}^\top \boldsymbol{y} \xrightarrow{?} \boldsymbol{\beta}^\star := \arg\min_{\boldsymbol{\beta}} \mathbb{E}[(\boldsymbol{y} - \boldsymbol{\beta}^\top \boldsymbol{x})^2]$

Consistency of estimator?

- classical low-dimensional setting (d fixed and $n \to \infty$): ✓
- classical high-dimensional setting ($d \gg n$) if sparsity [14]: ✓
- n, d, m are comparably large: ✗

for what sample size, and what data distributions, the estimator can generalize well?

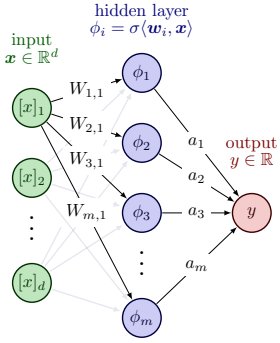
Problem settings



random features mapping [11]:

$$\varphi(\boldsymbol{x}) := \frac{1}{\sqrt{m}}\sigma\left(\frac{\boldsymbol{W}\boldsymbol{x}}{\sqrt{d}}\right), \quad W_{ij} \sim \mathcal{N}(0,1)$$

Problem settings



random features mapping [11]:

$$\varphi(\boldsymbol{x}) := \frac{1}{\sqrt{m}}\sigma\left(\frac{\boldsymbol{W}\boldsymbol{x}}{\sqrt{d}}\right), \quad W_{ij} \sim \mathcal{N}(0,1)$$

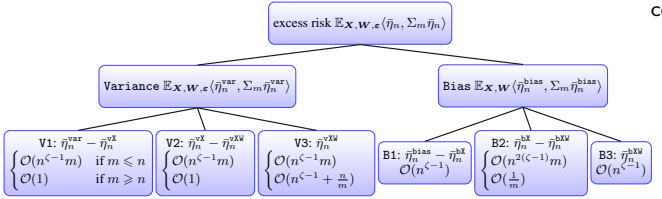
adaptive step-size SGD with one-pass iterate-averaging

$$\boldsymbol{a}_t = \boldsymbol{a}_{t-1} + \gamma_t[y_t - \langle \boldsymbol{a}_{t-1}, \varphi(\boldsymbol{x}_t)\rangle]\varphi(\boldsymbol{x}_t), \qquad t = 1, 2, \dots n\,.$$

- adaptive step-size: $\gamma_t := \gamma_0 t^{-\zeta}, \zeta \in [0, 1)$
- iterate-averaging: $\bar{\boldsymbol{a}}_n := \frac{1}{n}\sum_{t=0}^{n-1} \boldsymbol{a}_t$

Results: Bias and variance decomposition

Theorem [LSC, NeurIPS22]: Under sub-Gaussian data, label noise with bounded variance, we have

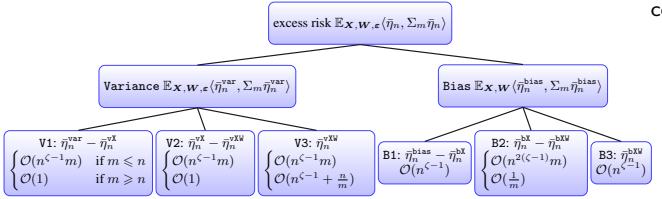


covariance operator: fast decay

- covariance operator: $\Sigma_m := \mathbb{E}_{\boldsymbol{x}}[\varphi(\boldsymbol{x}) \otimes \varphi(\boldsymbol{x})]$ (r.v.)
- expected covariance operator: $\widetilde{\Sigma}_m := \mathbb{E}_{\boldsymbol{x},\boldsymbol{W}}[\varphi(\boldsymbol{x}) \otimes \varphi(\boldsymbol{x})]$

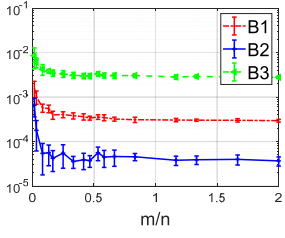
Results: Bias and variance decomposition

Theorem [LSC, NeurIPS22]: Under sub-Gaussian data, label noise with bounded variance, we have

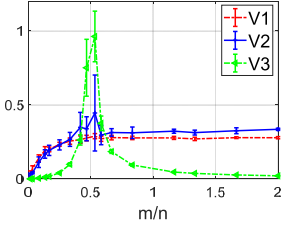


covariance operator: fast decay

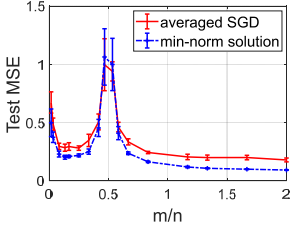
- covariance operator: $\Sigma_m := \mathbb{E}_{\boldsymbol{x}}[\varphi(\boldsymbol{x}) \otimes \varphi(\boldsymbol{x})]$ (r.v.)
- expected covariance operator: $\widetilde{\Sigma}_m := \mathbb{E}_{\boldsymbol{x},\boldsymbol{W}}[\varphi(\boldsymbol{x}) \otimes \varphi(\boldsymbol{x})]$



(a) Bias $\lesssim$ B1 + B2 + B3

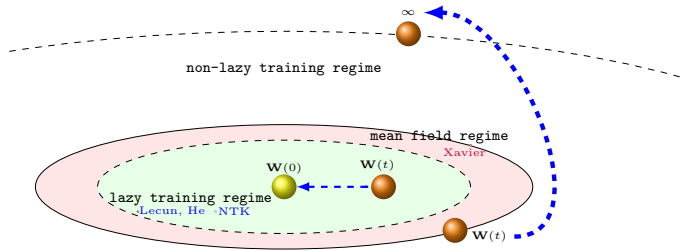


(b) Variance $\lesssim$ V1 + V2 + V3



(c) excess risk

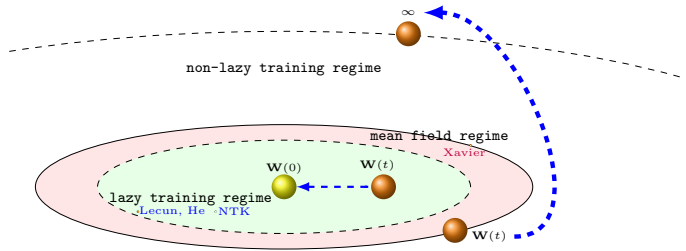
From kernel methods to neural networks



kernel methods

- high dimensional kernel methods: only learn linear function! [12]
- cannot efficiently approximate non-smooth functions

From kernel methods to neural networks



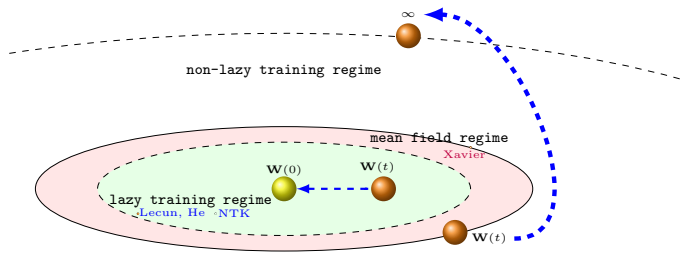
kernel methods

- high dimensional kernel methods: only learn linear function! [12]
- cannot efficiently approximate non-smooth functions

Follow-up:

- function space: function space for neural networks, e.g., Barron space [15, 16]
- benign overfitting: Benign overfitting in deep neural networks under lazy training [ZLCLC, ICML23]

From kernel methods to neural networks



kernel methods

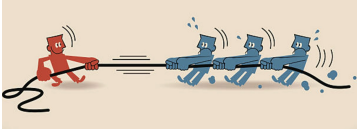
- high dimensional kernel methods: only learn linear function! [12]
- cannot efficiently approximate non-smooth functions

Follow-up:

- function space: function space for neural networks, e.g., Barron space [15, 16]
- benign overfitting: Benign overfitting in deep neural networks under lazy training [ZLCLC, ICML23]
- Benign overfitting leads to huge sensitivity! [4]

Over-parameterization helps or hurts robustness?

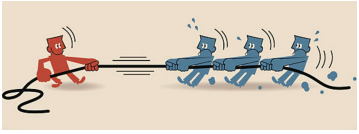
Helps! [17]



Hurts! [18, 19, 20]

Over-parameterization helps or hurts robustness?

Helps! [17]

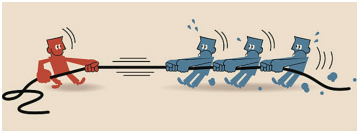


Hurts! [18, 19, 20]

- initialization (e.g., lazy training, non-lazy training)
- architecture (e.g., width, depth)

Over-parameterization helps or hurts robustness?

Helps! [17]



Hurts! [18, 19, 20]

- initialization (e.g., lazy training, non-lazy training)
- architecture (e.g., width, depth)

Definition (perturbation stability)

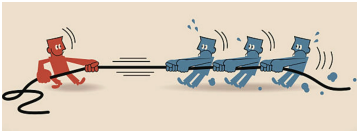
The perturbation stability of a ReLU DNN $f(\boldsymbol{x}; \boldsymbol{W})$ is

$$\mathscr{P}(f, \epsilon) = \mathbb{E}_{\boldsymbol{x}, \hat{\boldsymbol{x}}, \boldsymbol{W}} \left\| \nabla_{\boldsymbol{x}} f(\boldsymbol{x}; \boldsymbol{W})^\top (\boldsymbol{x} - \hat{\boldsymbol{x}}) \right\|_2, \quad \hat{\boldsymbol{x}} \sim \mathsf{Unif}(\mathbb{B}(\epsilon, \boldsymbol{x})),$$

where ϵ is the perturbation radius.

Over-parameterization helps or hurts robustness?

Helps! [17]



Hurts! [18, 19, 20]

- initialization (e.g., lazy training, non-lazy training)
- architecture (e.g., width, depth)

Definition (perturbation stability: lazy training regime)

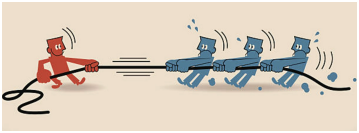
The perturbation stability of a ReLU DNN $f(\boldsymbol{x}; \boldsymbol{W})$ is

$$\mathscr{P}(f, \epsilon) = \mathbb{E}_{\boldsymbol{x}, \hat{\boldsymbol{x}}, \boldsymbol{W}(0)} \left\| \nabla_{\boldsymbol{x}} f(\boldsymbol{x}; \boldsymbol{W})^{\top} (\boldsymbol{x} - \hat{\boldsymbol{x}}) \right\|_2, \quad \hat{\boldsymbol{x}} \sim \mathsf{Unif}(\mathbb{B}(\epsilon, \boldsymbol{x})),$$

where ϵ is the perturbation radius.

Over-parameterization helps or hurts robustness?

Helps! [17]



Hurts! [18, 19, 20]

- initialization (e.g., lazy training, non-lazy training)
- architecture (e.g., width, depth)

Definition (perturbation stability: non-lazy training regime)

The perturbation stability of a ReLU DNN $f(\boldsymbol{x}; \boldsymbol{W})$ is

$$\mathscr{P}(f, \epsilon) = \mathbb{E}_{\boldsymbol{x}, \hat{\boldsymbol{x}}} \left\| \nabla_{\boldsymbol{x}} f(\boldsymbol{x}; \boldsymbol{W})^{\top} (\boldsymbol{x} - \hat{\boldsymbol{x}}) \right\|_2, \quad \hat{\boldsymbol{x}} \sim \mathsf{Unif}(\mathbb{B}(\epsilon, \boldsymbol{x})),$$

where ϵ is the perturbation radius.

Main results (Lazy-training regime)

Theorem[ZLCC, NeurIPS22]: $\cdot \lesssim \mathrm{Func}(m, L, \beta)$

Assumption	Initialization	Our bound for $\mathscr{P}(f,\epsilon)/\epsilon$	Trend of width m [1]	Trend of depth L [1]
$\|\boldsymbol{x}\|_2 = 1$	LeCun initialization	$\left(\sqrt{\frac{L^3 m}{d}} e^{-m/L^3} + \sqrt{\frac{1}{d}}\right)(\frac{\sqrt{2}}{2})^{L-2}$	↗ ↘	↘
	He initialization	$\sqrt{\frac{L^3 m}{d}} e^{-m/L^3} + \sqrt{\frac{1}{d}}$	↗ ↘	↗
	NTK initialization	$\sqrt{\frac{L^3 m}{d}} e^{-m/L^3} + 1$	↗ ↘	↗

[1] The larger perturbation stability means worse average robustness.

Takeaway messages: **the good (width), the bad (depth), the ugly (initialization)**

Main results (Lazy-training regime)

Theorem[ZLCC, NeurIPS22]: $\cdot \lesssim \mathrm{Func}(m, L, \beta)$

Assumption	Initialization	Our bound for $\mathscr{P}(f,\epsilon)/\epsilon$	Trend of width m [1]	Trend of depth L [1]
$\|\boldsymbol{x}\|_2 = 1$	LeCun initialization	$\left(\sqrt{\frac{L^3 m}{d}} e^{-m/L^3} + \sqrt{\frac{1}{d}}\right)(\frac{\sqrt{2}}{2})^{L-2}$	↗ ↘	↘
	He initialization	$\sqrt{\frac{L^3 m}{d}} e^{-m/L^3} + \sqrt{\frac{1}{d}}$	↗ ↘	↗
	NTK initialization	$\sqrt{\frac{L^3 m}{d}} e^{-m/L^3} + 1$	↗ ↘	↗

[1] The larger perturbation stability means worse average robustness.

Takeaway messages: **the good (width), the bad (depth), the ugly (initialization)**

- width **helps** robustness in the over-parameterized regime

Main results (Lazy-training regime)

Theorem[ZLCC, NeurIPS22]: $\cdot \lesssim \mathrm{Func}(m, L, \beta)$

Assumption	Initialization	Our bound for $\mathscr{P}(f,\epsilon)/\epsilon$	Trend of width m [1]	Trend of depth L [1]
$\|\boldsymbol{x}\|_2 = 1$	LeCun initialization	$\left(\sqrt{\frac{L^3 m}{d}} e^{-m/L^3} + \sqrt{\frac{1}{d}}\right)(\frac{\sqrt{2}}{2})^{L-2}$	↗ ↘	↘
	He initialization	$\sqrt{\frac{L^3 m}{d}} e^{-m/L^3} + \sqrt{\frac{1}{d}}$	↗ ↘	↗
	NTK initialization	$\sqrt{\frac{L^3 m}{d}} e^{-m/L^3} + 1$	↗ ↘	↗

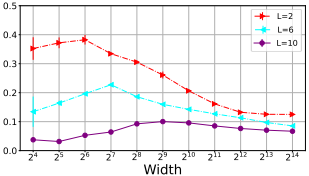
[1] The larger perturbation stability means worse average robustness.

Takeaway messages: **the good (width), the bad (depth), the ugly (initialization)**

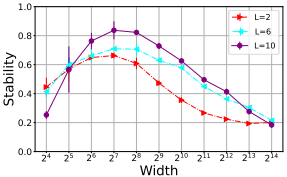
- width **helps** robustness in the over-parameterized regime
- depth **helps** robustness in LeCun initialization but **hurts** robustness in He/NTK initialization

Experiments: robustness under lazy-training regime

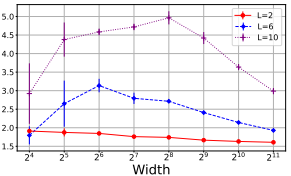
Metrics	Ours (NTK initialization)	[19]	[20]
$\mathscr{P}(f,\epsilon)/\epsilon$	$\sqrt{\frac{L^3 m}{d}} e^{-m/L^3} + 1$	$L^2 m^{1/3} \sqrt{\log m} + \sqrt{mL}$	$2^{\frac{3L-5}{2}} \sqrt{L}$



(a) LeCun initialization



(b) He initialization



(c) NTK initialization

Proof sketch in robustness

$$
\begin{aligned}
\mathscr{P}(\boldsymbol{f}, \epsilon) &:= \mathbb{E}_{\boldsymbol{x},\hat{\boldsymbol{x}},\boldsymbol{W}} \left\|\nabla_{\boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x})(\boldsymbol{x}-\hat{\boldsymbol{x}})\right\|_2 \\
&\leq \mathbb{E}_{\boldsymbol{x},\hat{\boldsymbol{x}},\boldsymbol{W}} \left\|\nabla_{\boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x})(\boldsymbol{x}-\hat{\boldsymbol{x}}) - \boldsymbol{W}_L \boldsymbol{D}_{L-1} \boldsymbol{W}_{L-1} \cdots \boldsymbol{D}_1 \boldsymbol{W}_1 (\boldsymbol{x}-\hat{\boldsymbol{x}})\right\|_2 + \mathbb{E}_{\boldsymbol{x},\hat{\boldsymbol{x}},\boldsymbol{W}} \left\|\boldsymbol{W}_L \boldsymbol{D}_{L-1} \boldsymbol{W}_{L-1} \cdots \boldsymbol{D}_1 \boldsymbol{W}_1 (\boldsymbol{x}-\hat{\boldsymbol{x}})\right\|_2 \\
&\leq \mathbb{E}_{\boldsymbol{x},\hat{\boldsymbol{x}},\boldsymbol{W}} \left\|[\nabla_{\boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{W}_L \boldsymbol{D}_{L-1} \boldsymbol{W}_{L-1} \cdots \boldsymbol{D}_1 \boldsymbol{W}_1](\boldsymbol{x}-\hat{\boldsymbol{x}})\right\|_2 + \sqrt{\mathbb{E}_{\boldsymbol{x},\hat{\boldsymbol{x}},\boldsymbol{W}} \left\|\boldsymbol{W}_L \underbrace{\boldsymbol{D}_{L-1} \boldsymbol{W}_{L-1} \cdots \boldsymbol{D}_1 \boldsymbol{W}_1 (\boldsymbol{x}-\hat{\boldsymbol{x}})}_{:=\boldsymbol{t}_L}\right\|_2^2} \\
&\lesssim \epsilon \left(\sqrt{L^3 m^2 \beta_1^2 \beta_L^2} e^{-m/L^3} + \sqrt{mo \beta_1^2 \beta_L^2} \right) \gamma^{L-2},
\end{aligned}
$$

where we use $\mathbb{E}_{\boldsymbol{W}} \frac{\|\boldsymbol{t}_l\|_2^2}{\left\|\boldsymbol{t}_{l-1}\right\|_2^2} = \gamma^2$ with $\gamma := \beta / \sqrt{\frac{2}{m}}$.

Main results (Non-lazy training regime)

A sufficient condition for DNNs

For large enough m and $m \gg d$, w.h.p, DNNs fall into non-lazy training regime if $\alpha \gg (m^{3/2} \sum_{i=1}^{L} \beta_i)^L$.

Remarks: $\circ$ $L = 2$, $\alpha = 1$, $\beta_1 = \beta_2 = \beta \sim \frac{1}{m^c}$ with $c > 1.5$

Main results (Non-lazy training regime)

A sufficient condition for DNNs

For large enough m and $m \gg d$, w.h.p, DNNs fall into non-lazy training regime if $\alpha \gg (m^{3/2} \sum_{i=1}^{L} \beta_i)^L$.

Remarks: ○ $L = 2$, $\alpha = 1$, $\beta_1 = \beta_2 = \beta \sim \frac{1}{m^c}$ with $c > 1.5$

Theorem (non-lazy training regime for two-layer NNs)

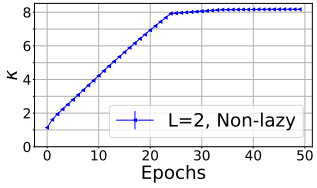
Under this setting with $m \gg n^2$ and standard assumptions, then

$$\textit{perturbation stability} \leq \widetilde{\mathcal{O}}\left(\frac{n}{m^{c+1.5}}\right), \ w.h.p$$

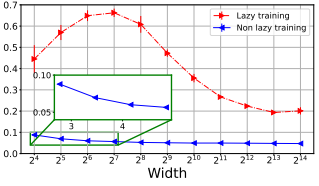
Remarks: ○ width helps robustness in the over-parameterized regime in both lazy/non-lazy training regime

Experiment: Non-lazy training regime

$$\text{lazy training ratio } \kappa := \frac{\sum_{l=1}^{L} \|\boldsymbol{W}_l(t) - \boldsymbol{W}_l(0)\|_{\mathrm{F}}}{\sum_{l=1}^{L} \|\boldsymbol{W}_l(0)\|_{\mathrm{F}}}$$



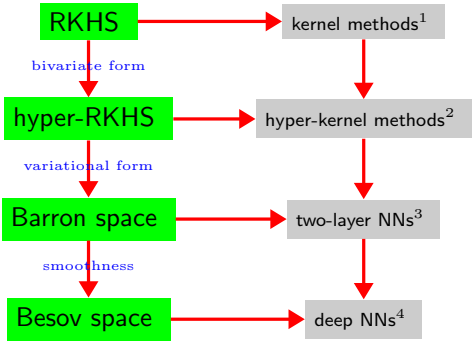
(a) lazy training ratio *vs.* epochs

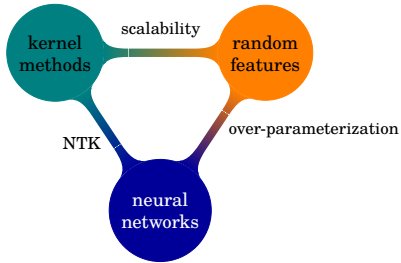


(b) perturbation stability

Conclusion(I) function spaces vs. models

Understanding from a function space perspective!





[1] [LHGYL, JMLR20; LHCS, TPAMI21; LLS, AISTATS21]

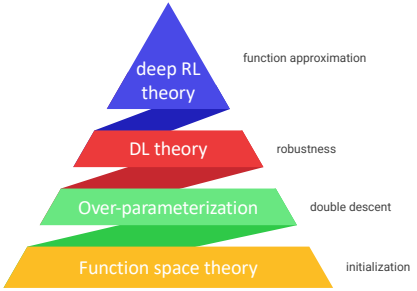
[2] [LSHYS, JMLR21]

[3] [LSC, NeurIPS22; LHCS, TPAMI22; LHCS, AISTATS21]

[4] [LVC, NeurIPS22; ZLCC, NeurIPS22; WZLCC, NeurIPS22, ZLCLC, ICML23]

Conclusion(II) the good, the bad, the ugly

	good	bad	ugly
kernel methods	analysis	performance	curse of dimensionality
neural networks	performance	analysis	over-parameterization
generalization	benign overfitting	catastrophic overfitting	model complexity
robustness	width	depth	initialization



- IEEE ICASSP 2023 Tutorial - "Neural networks: the good, the bad, and the ugly"
- CVPR 2023 Tutorial - "Deep learning theory for computer vision"

Thanks for your attention!

Q & A

my homepage www.lfhsgre.org for more information!

References I

[1] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals.
Understanding deep learning (still) requires rethinking generalization.
Communications of the ACM, 64(3):107–115, 2021.
(Cited on page 5.)

[2] Zuxuan Wu, Ser-Nam Lim, Larry S Davis, and Tom Goldstein.
Making an invisibility cloak: Real world adversarial attacks on object detectors.
In *European Conference on Computer Vision*, pages 1–17. Springer, 2020.
(Cited on pages 6 and 7.)

[3] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song.
Robust physical-world attacks on deep learning visual classification.
In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1625–1634, 2018.
(Cited on pages 6 and 7.)

[4] Peter L. Bartlett, Philip M. Long, Gábor Lugosi, and Alexander Tsigler.
Benign overfitting in linear regression.
the National Academy of Sciences, 2020.
(Cited on pages 8, 9, 10, 11, 12, 13, 14, 30, 31, and 32.)

References II

[5] Niladri S Chatterji and Philip M Long.
Foolish crowds support benign overfitting.
Journal of Machine Learning Research, 23(125):1–12, 2022.
(Cited on pages 13 and 14.)

[6] Spencer Frei, Niladri S Chatterji, and Peter Bartlett.
Benign overfitting without linearity: Neural network classifiers trained by gradient descent for noisy linear data.
In *Conference on Learning Theory*, pages 2668–2703. PMLR, 2022.
(Cited on pages 13 and 14.)

[7] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal.
Reconciling modern machine-learning practice and the classical bias–variance trade-off.
the National Academy of Sciences, 116(32):15849–15854, 2019.
(Cited on pages 13 and 14.)

[8] Arthur Jacot, Franck Gabriel, and Clément Hongler.
Neural tangent kernel: Convergence and generalization in neural networks.
In *Advances in Neural Information Processing Systems*, pages 8571–8580, 2018.
(Cited on pages 16, 17, 19, 20, and 21.)

References III

[9] Lenaic Chizat, Edouard Oyallon, and Francis Bach.
On lazy training in differentiable programming.
In *Advances in Neural Information Processing Systems*, pages 2933–2943, 2019.
(Cited on pages 16, 17, 19, and 20.)

[10] Tao Luo, Zhi-Qin John Xu, Zheng Ma, and Yaoyu Zhang.
Phase diagram for two-layer relu neural networks at infinite-width limit.
Journal of Machine Learning Research, 22(71):1–47, 2021.
(Cited on pages 16, 17, 19, and 20.)

[11] Ali Rahimi and Benjamin Recht.
Random features for large-scale kernel machines.
In *Advances in Neural Information Processing Systems*, pages 1177–1184, 2007.
(Cited on pages 21, 26, and 27.)

[12] Noureddine El Karoui.
The spectrum of kernel random matrices.
Annals of Statistics, 38(1):1–50, 2010.
(Cited on pages 22, 23, 24, 25, 30, 31, and 32.)

References IV

[13] Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J Tibshirani.
Surprises in high-dimensional ridgeless least squares interpolation.
Annals of Statistics, 50(2):949–986, 2022.
(Cited on pages 22, 23, 24, and 25.)

[14] Emmanuel J Candes and Terence Tao.
Decoding by linear programming.
IEEE Transactions on Information Theory, 51(12):4203–4215, 2005.
(Cited on pages 24 and 25.)

[15] Weinan E, Chao Ma, and Lei Wu.
The barron space and the flow-induced function spaces for neural network models.
Constructive Approximation, pages 1–38, 2021.
(Cited on pages 30, 31, and 32.)

[16] Rahul Parhi and Robert D Nowak.
Near-minimax optimal estimation with shallow ReLU neural networks.
IEEE Transactions on Information Theory, 2022.
(Cited on pages 30, 31, and 32.)

References V

[17] Sébastien Bubeck and Mark Sellke.
A universal law of robustness via isoperimetry.
In *Advances in Neural Information Processing Systems*, pages 28811–28822, 2021.
(Cited on pages 33, 34, 35, 36, and 37.)

[18] Hamed Hassani and Adel Javanmard.
The curse of overparametrization in adversarial training: Precise analysis of robust generalization for random features regression.
arXiv preprint arXiv:2201.05149, 2022.
(Cited on pages 33, 34, 35, 36, and 37.)

[19] Boxi Wu, Jinghui Chen, Deng Cai, Xiaofei He, and Quanquan Gu.
Do wider neural networks really help adversarial robustness?
In *Advances in Neural Information Processing Systems*, pages 7054–7067, 2021.
(Cited on pages 33, 34, 35, 36, 37, and 41.)

[20] Hanxun Huang, Yisen Wang, Sarah Erfani, Quanquan Gu, James Bailey, and Xingjun Ma.
Exploring architectural ingredients of adversarially robust deep neural networks.
In *Advances in Neural Information Processing Systems*, pages 5545–5559, 2021.
(Cited on pages 33, 34, 35, 36, 37, and 41.)