Generalization theory of Deep Learning

Yiqiao Zhong, Department of Statistics, UW Madison

CS 762, Oct 20, 2022

Some notations

- Training data $S = (\boldsymbol{x}_i, y_i)_{i=1}^n$, input x_i is d-dim vector, y_i label (or real value)
- Neural network (NN) $f_{\boldsymbol{\theta}}$: $R^d \to R^K$(or R), e.g., feed-forward NN

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \mathbf{W}_L \sigma(\cdots \sigma(\mathbf{W}_2\, \sigma(\mathbf{W}_1 \boldsymbol{x} + \boldsymbol{b}_1) + \boldsymbol{b}_2) + \boldsymbol{b}_L)$$

where σ is activation (e.g., ReLU); $\boldsymbol{\theta}$ contains all trainable parameters

- Loss function $L(y, f_{\boldsymbol{\theta}}(\boldsymbol{x}))$, Empirical Risk Minimization (ERM)

$$\widehat{\boldsymbol{\theta}} \in \operatorname{argmin}_{\boldsymbol{\theta}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)) =: R_n(\boldsymbol{\theta})$$

- Train loss $R_n(\widehat{\boldsymbol{\theta}})$, **train error**: ratio of misclassification on S
- On test data $T = (\boldsymbol{x}_i, y_i)_{i=1}^{n'}$, evaluate $\widehat{\boldsymbol{\theta}}$, **test error** (sometimes generalization error): ratio of misclassification on T. Often $n' = \infty$ in analysis
- Disclaimer: very incomplete references; check [Bartlett, Montanari, Rakhlin, Deep learning: a statistical view, 2021]

The generalization puzzle

Bias-variance tradeoff

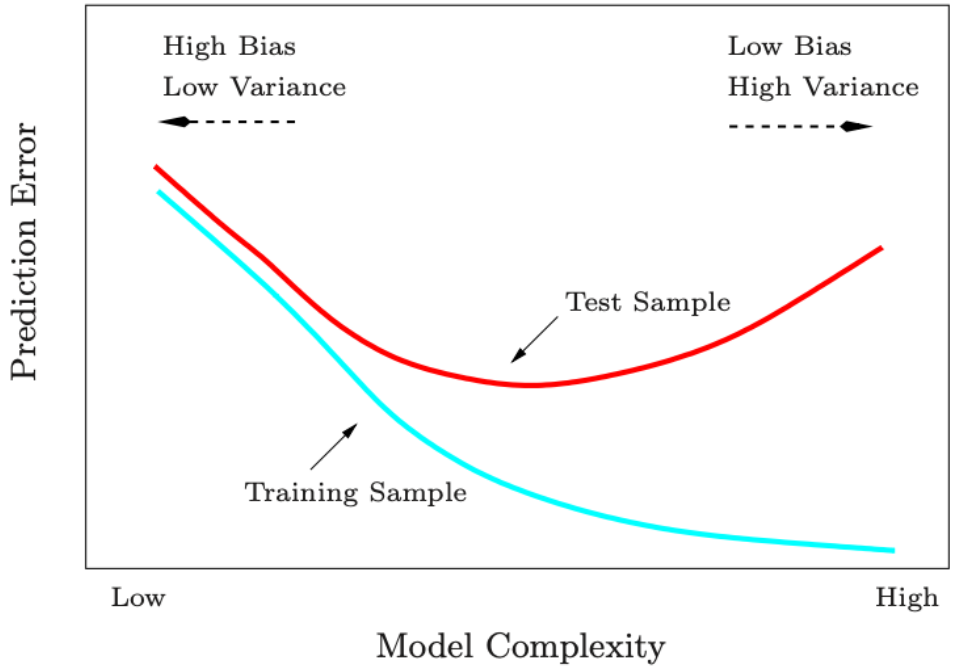


FIGURE 2.11. *Test and training error as a function of model complexity.*

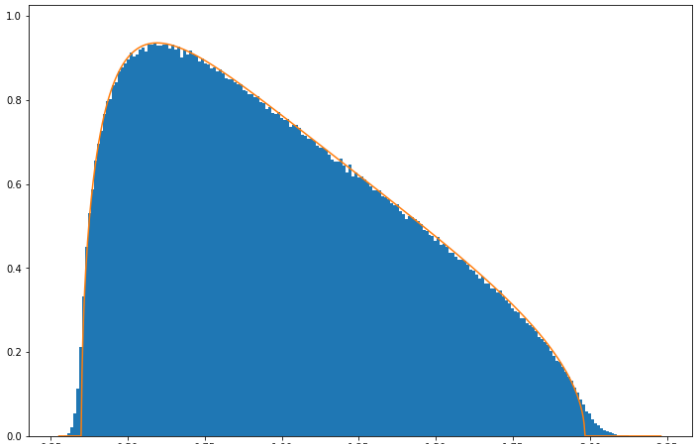
- Generally holds for many statistical models
- Classical solution to high-complexity models: regularize!

Source: ESL

Why and how regularizing high complexity model

- Consider linear ridge regression. Denote $n \times d$ data matrix $\boldsymbol{X}$. Solve

$$\min_{\boldsymbol{\theta}} \frac{1}{n}\|\boldsymbol{y} - \mathbf{X}\boldsymbol{\theta}\|_2^2 + \lambda\|\boldsymbol{\theta}\|_2^2$$



- Using SVD : $n^{-1/2}\boldsymbol{X} = \boldsymbol{U\Sigma V^T}$

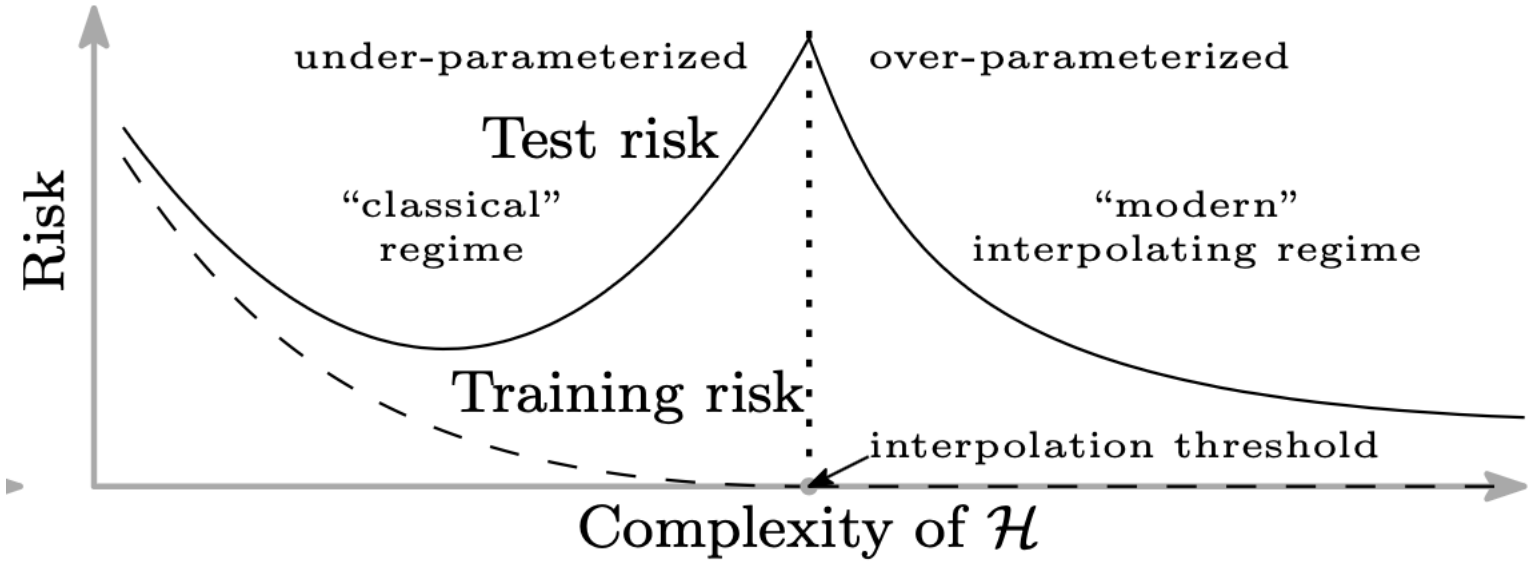
$$\widehat{\boldsymbol{\theta}} = \left(\frac{1}{n}\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I}_d\right)^{-1}\frac{1}{n}\mathbf{X}^\top\boldsymbol{y} = \frac{1}{\sqrt{n}}\sum_{j=1}^{d}\boldsymbol{u}_j\frac{\sigma_j}{\sigma_j^2+\lambda}\boldsymbol{v}_j^\top\boldsymbol{y}$$

- Test error generally $O(d/n)$ if $d \ll n$
- Worse still, if d is close to n, **huge variance** in $\widehat{\boldsymbol{\theta}}$ without regularization. (MP law)
- Solution: need **large $\boldsymbol{\lambda}$** if d is large.

Successful stories of regularization are everywhere:

- If signal is a sparse vector, use L_1 regularization $\| \boldsymbol{\theta} \|_1$, called LASSO
- If signal is a low-rank matrix, use nuclear-norm regularization $\| \boldsymbol{\theta} \|_*$

But wait…double descent ?!

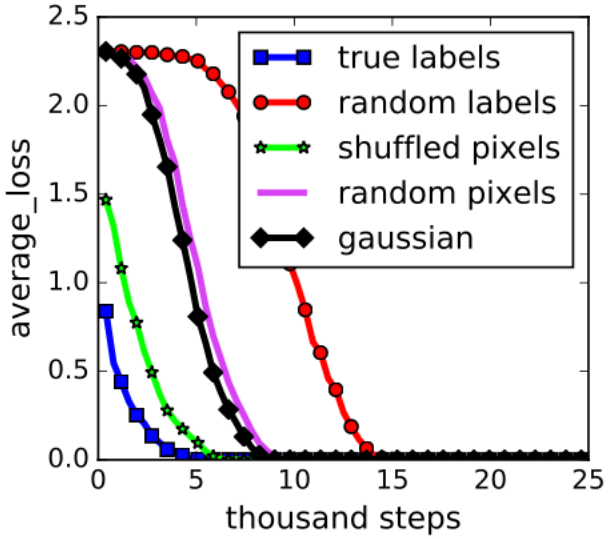


New questions for ML/statistics:

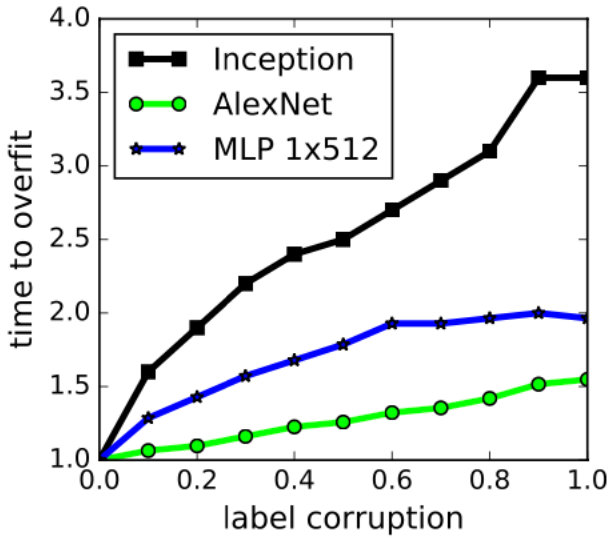
1. When and why this happens?
2. When second descent better? Do we need regularization?
3. Lessons for architecture & algorithm design?

Source: Belkin et al, 2019

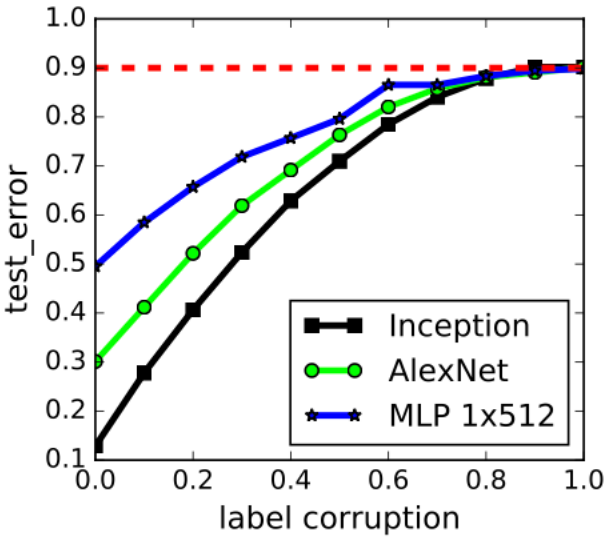
Need understanding beyond interpolation



(a) learning curves



(b) convergence slowdown



(c) generalization error growth

Source: Zhang et al, 2019

Search for implicit bias

Implicit bias

- Space of interpolating solutions (train error/loss is zero) may be large, but (stochastic) gradient descent (GD) converges to one with good generalization performance
- Proof-of-concept in overparametrized linear regression:

Proposition (HMRT, 2018). Initialize $\boldsymbol{\beta}^{(0)} = \mathbf{0}$, and consider gradient descent on $L_n(\boldsymbol{\beta}) = n^{-1}\|\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta}\|^2$,

$$\boldsymbol{\beta}^{(k)} = \boldsymbol{\beta}^{(k-1)} - \eta \nabla_{\boldsymbol{\beta}} L_n(\boldsymbol{\beta}), \qquad k = 1, 2, 3, \ldots,$$

where $\eta > 0$ is sufficiently small. Then $\lim_{k\to\infty} \boldsymbol{\beta}^{(k)} = \widehat{\boldsymbol{\beta}}$, where $\widehat{\boldsymbol{\beta}}$ is the minimum-norm interpolator:

$$\widehat{\boldsymbol{\beta}} = \operatorname{argmin}\Big\{ \|\boldsymbol{\beta}\|_2 : \boldsymbol{\beta} \text{ minimizes } L_n(\boldsymbol{\beta}) \Big\} = \left(\mathbf{X}^\top \mathbf{X}\right)^+ \mathbf{X}^\top \boldsymbol{y}.$$

Note that $\left(\mathbf{X}^\top \mathbf{X}\right)^+$ denotes the Moore-Penrose pseudoinverse. In particular, if $\mathbf{X}\mathbf{X}^\top$ is invertible, then $\widehat{\beta} = \mathbf{X}^\top (\mathbf{X}\mathbf{X}^\top)^{-1} \boldsymbol{y}$ is an interpolator.

Implicit bias for classifying separable data

- Classification setting: for *linearly separable data* we can achieve zero train error using a linear classifier.

Theorem (Soudry, et al. 2018)**.** Consider the logistic loss and any linearly separable data. From any initializer $\boldsymbol{\beta}^{(0)} \in \mathbb{R}^d$, the gradient descent iterate $\boldsymbol{\beta}^{(k+1)} = \boldsymbol{\beta}^{(k)} - \eta \nabla L_n(\boldsymbol{\beta}^{(k)})$ satisfies

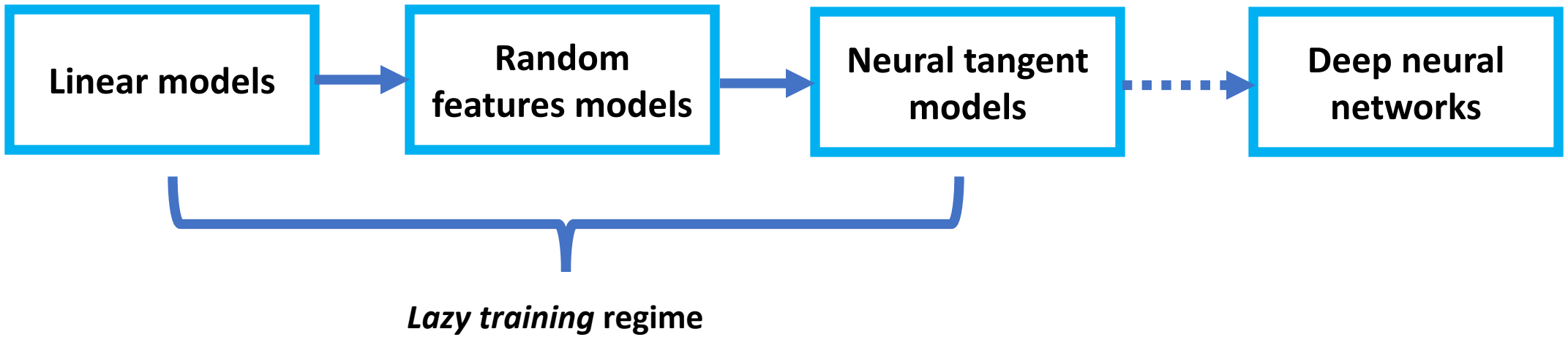
$$\boldsymbol{\beta}^{(k)} = \widehat{\boldsymbol{\beta}} \log k + \boldsymbol{\Delta}^{(k)}$$

where residual $\|\boldsymbol{\Delta}^{(k)}\|_2 = O(\log \log k)$ and $\widehat{\boldsymbol{\beta}}$ is the max-margin solution

$$\widehat{\boldsymbol{\beta}} = \operatorname{argmin}_{\boldsymbol{\beta}} \|\boldsymbol{\beta}\|_2^2 \quad \text{s.t. } \boldsymbol{\beta}^\top \boldsymbol{x}_i \geq 1, \text{ for all } i = 1, 2, \ldots, n.$$

- Gradient descent favors “small-norm” solution (at least in certain settings)
- Search for implicit bias: multiple linear deep network [Moroshko et al. 20], linear convolution network [GLSS18], one-hidden-layer ReLU network [NTS15], etc.
- Q: What is the generalization error of these solutions?

The path to realism (or not?)



- In lazy training regime [OCB19], models are linear in parameters [HMRT18, MM19, MRSY19, MZ20]
- Test error can be calculated with idealized assumptions on data, rigorously justifying double descent

Neural tangent model

- Key insight: when network width is infinite (or very large), the GD or SGD dynamics is given by (or approximated) by linearized local models---known as *neural tangent kernel* (NTK) models [JGH18, DZPS19, AZLS19, COB19]
- A simple example: one-hidden-layer NN:

$$f(\boldsymbol{x};\boldsymbol{\theta}) = \frac{1}{N}\sum_{k=1}^{N} a_k \sigma(\boldsymbol{w}_k^\top \boldsymbol{x})$$

- Initialize from $\boldsymbol{\theta}_0 = (\boldsymbol{a}_0, \boldsymbol{W}_0)$, do **Taylor expansion**:

$$\begin{aligned}
&\frac{1}{\varepsilon} f(\boldsymbol{x}; \boldsymbol{a}_0 + \varepsilon \boldsymbol{a}, \mathbf{W}_0 + \varepsilon \mathbf{W}) \\
\approx\ &\underbrace{\frac{1}{\varepsilon} f(\boldsymbol{x}; \boldsymbol{a}_0, \mathbf{W}_0)}_{\text{initialization}} + \underbrace{\sum_{k=1}^{N} a_k \sigma(\langle \boldsymbol{w}_{0,k}, \boldsymbol{x}\rangle)}_{\text{random features model}} + \underbrace{\sum_{k=1}^{N} a_{0,k} \langle \boldsymbol{w}_k, \boldsymbol{x}\rangle \sigma'(\langle \boldsymbol{w}_{0,k}, \boldsymbol{x}\rangle)}_{\text{neural tangent model}}
\end{aligned}$$

Why NTK makes sense?

Consider residuals at time t: let $\boldsymbol{r}_t = \boldsymbol{y} - f(\boldsymbol{x}; \boldsymbol{\theta}_t) \in \mathbb{R}^n$, then running gradient flow (GF) gives

$$\frac{d}{dt}\boldsymbol{r}_t = -\mathbf{K}_t \boldsymbol{r}_t, \qquad \text{where } [\mathbf{K}_t]_{ij} = \langle \nabla_\theta f(\boldsymbol{x}_i; \boldsymbol{\theta}_t), \nabla_\theta f(\boldsymbol{x}_j; \boldsymbol{\theta}_t) \rangle$$

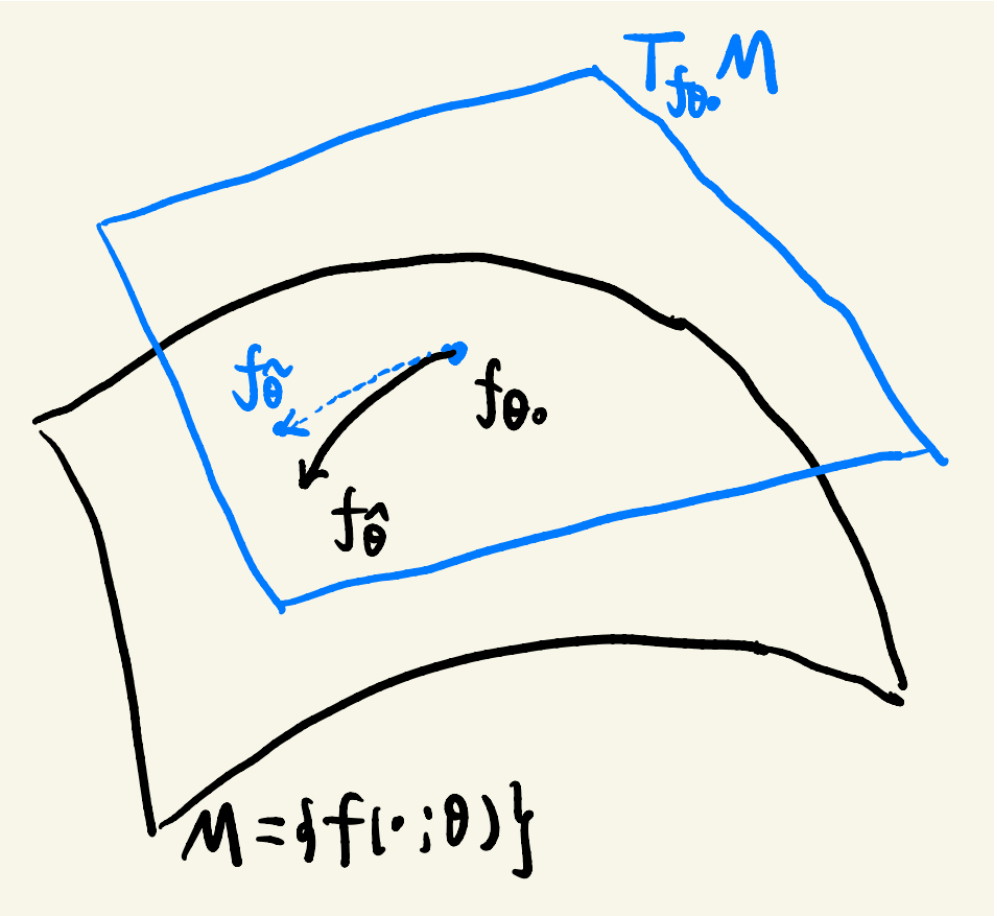
If $\mathbf{K}_t \approx \mathbf{K}_0$ for all t (guaranteed when width is large), then

$$(i) \qquad \frac{d}{dt}\left\|\boldsymbol{r}_t\right\|_2^2 = -2\boldsymbol{r}_t^\top \mathbf{K}_t \boldsymbol{r}_t \lesssim -2\lambda_{\min}(\mathbf{K}_0)\left\|\boldsymbol{r}_t\right\|_2^2$$

$$\implies \qquad \text{linear convergence rate}$$

$$(ii) \qquad \boldsymbol{r}_t \approx \widetilde{\boldsymbol{r}}_t \qquad \text{where } \widetilde{\boldsymbol{r}}_t \text{ comes from GF}$$

$$\text{on the loss } \widetilde{L}_n(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - f_{\text{lin}}(\boldsymbol{x}_i; \boldsymbol{\theta})\right)^2$$



- Q: under NTK, what is the generalization error?

Insights from statistics

Overparametrized linear models

- Consider the one-hidden-layer NTK model. We have *NT features:*

$$\nabla_{\boldsymbol{w}} f(\boldsymbol{x};\boldsymbol{\theta}_0) = [a_1\sigma'(\boldsymbol{x}^\top \boldsymbol{w}_{0,1})\boldsymbol{x}^\top, \ldots, a_N\sigma'(\boldsymbol{x}^\top \boldsymbol{w}_{0,N})\boldsymbol{x}^\top] \in \mathbb{R}^{Nd}$$

- A useful simplification: NT features have complicated dependence, why not assume that we have $\widetilde{\boldsymbol{x}}_i \sim N(\boldsymbol{0}, \boldsymbol{\Sigma})$, prediction function is $\widetilde{\boldsymbol{x}}^T \widehat{\boldsymbol{\theta}}$. By abuse of notations, just write $\boldsymbol{x}_i$.

Assumption. Suppose $\boldsymbol{z} = \boldsymbol{\Sigma}^{-1/2}\boldsymbol{x}$ is 1-subgaussian.
WLOG, assume $\boldsymbol{\Sigma} = \mathrm{diag}(\lambda_1, \ldots, \lambda_d)$ where $\lambda_1 \geq \cdots \geq \lambda_d$.

- Key insight:

$$\langle \boldsymbol{x}, \widehat{\boldsymbol{\theta}} \rangle = \underbrace{\langle \boldsymbol{x}_{\leq k}, \widehat{\boldsymbol{\theta}}_{\leq k} \rangle}_{\text{prediction part}} + \underbrace{\langle \boldsymbol{x}_{>k}, \widehat{\boldsymbol{\theta}}_{>k} \rangle}_{\text{interpolation part}}$$

Decomposing features

- Regression setting. Data $\mathbf{X} = [\mathbf{X}_{\leq k}, \mathbf{X}_{>k}]$ of size $n \times d$, where $d > n$

$$\mathbf{X}\mathbf{X}^\top = \mathbf{X}_{\leq k}\mathbf{X}_{\leq k}^\top + \mathbf{X}_{>k}\mathbf{X}_{>k}^\top$$

- A seemingly bold assumption: $\mathbf{X}_{>k}\mathbf{X}^T{}_{>k} \approx \gamma\, \mathbf{I}_n$
- Heuristic justification: features are divided into "important" ones ($\leq k$) and "not important" ones ($> k$); the latter is similar to pure noise

$$\widehat{\boldsymbol{\theta}}_{\leq k} = \operatorname{argmin}_{\boldsymbol{\theta} \in \mathbb{R}^k} \left\| \boldsymbol{y} - \mathbf{X}_{\leq k}\boldsymbol{\theta} \right\|_2^2 + \gamma \|\boldsymbol{\theta}\|_2^2$$

- Equivalent to **ridge regression**!

Implicit regularization

- Parameter γ controls the amount of regularization
- Turning heuristics into rigorous arguments. For general $(\lambda_j)_{j\le d}$, define *effective rank:*

$$r_k(\mathbf{\Sigma}) = \frac{\sum_{i>k}\lambda_i}{\lambda_{k+1}}.$$

- Concentration results can show:

$$r_k(\mathbf{\Sigma}) \ge bn \Longrightarrow \lambda_j(\mathbf{X}_{>k}\mathbf{X}_{>k}^\top) \in [\gamma/c, c\gamma] \text{ where } \gamma = \sum_{j>k}\lambda_j$$

- Find a sweet spot for k so that: $\boldsymbol{x}_{\le k}$ captures almost all information while $\boldsymbol{x}_{>k}$ is similar to noise. Called *effective dimension*.

A look at the theorems

Theorem 4.4. *Fix $\delta < 1/2$. Under Assumption 4.2, suppose for some k the condition number of $\boldsymbol{X}_{>k}\boldsymbol{X}_{>k}^{\mathsf{T}}$ is at most κ with probability at least $1-\delta$. Then*

$$\widehat{\mathrm{VAR}} \lesssim \sigma_\xi^2 \kappa^2 \log\left(\frac{1}{\delta}\right)\left(\frac{k}{n} + \frac{n\sum_{i>k}\lambda_i^2}{(\sum_{i>k}\lambda_i)^2}\right) \tag{44}$$

with probability at least $1-2\delta$.

Theorem 4.5. *Under the assumptions of Theorem 4.4, for $n \gtrsim \log(1/\delta)$, with probability at least $1-2\delta$,*

$$\widehat{\mathrm{BIAS}}^2 \lesssim \kappa^4 \left[\left\|\boldsymbol{\theta}^*_{\leq k}\right\|^2_{\boldsymbol{\Sigma}^{-1}_{\leq k}} \left(\frac{\sum_{i>k}\lambda_i}{n}\right)^2 + \left\|\boldsymbol{\theta}^*_{>k}\right\|^2_{\boldsymbol{\Sigma}_{>k}}\right]. \tag{48}$$

- Upper bounds tight up to constants [Tsigler, Bartlett, 2020]
- Bias and variance vanish under suitable decay of eigenvalues [TB20], empirically checked [WHS22]

Is linear model naïve?

- Consider the NT features:

$$\underbrace{\sigma'(\langle \boldsymbol{x}, \boldsymbol{w}_k \rangle)\boldsymbol{x}}_{\text{NT feature}} = \underbrace{\sigma'_{\leq \ell}(\langle \boldsymbol{x}, \boldsymbol{w}_k \rangle)\boldsymbol{x}}_{\text{effective fitting}} + \underbrace{\sigma'_{>\ell}(\langle \boldsymbol{x}, \boldsymbol{w}_k \rangle)\boldsymbol{x}}_{\text{noise}}$$

- The spirit is the same. Stacking NT features into $n \times (Nd)$ matrix Φ . Assume isotropic data $\boldsymbol{x}_i \sim N(\mathbf{0}, \boldsymbol{I}_d)$.

$$\begin{aligned}
\boldsymbol{\Phi}\boldsymbol{\Phi}^\top &\approx \boldsymbol{\Phi}_{\leq k}\boldsymbol{\Phi}_{\leq k}^\top + \boldsymbol{\Phi}_{>k}\boldsymbol{\Phi}_{>k}^\top && \text{(cross term negligible)} \\
&\approx \boldsymbol{\Phi}_{\leq k}\boldsymbol{\Phi}_{\leq k}^\top + \|\sigma'_{>k}\|_{L^2}^2 \cdot \mathbf{I}_n && \text{(high-degree term concentrates)}
\end{aligned}$$

- *Self-induced regularization*: **nonlinearity** of activation helps!

A general generalization result for 2-layer NTK

- [Montanari, Zhong, 2020] Suppose $d^k \ll n \ll d^{k+1}$, isotropic input data. general target function $f_* \in L_2\left(S^{d-1}\right)$. As long as network width N satisfies $Nd \gg n$ (overparametrization), then with high probability,

$$R_{\mathrm{NT}}(f;\lambda) = R_{\mathrm{KRR}}(f;\lambda) + O\left(\sqrt{\frac{n}{Nd}}\right)$$

$$= R_{\mathrm{PRR}}(f;\lambda + \|\sigma'_{>k}\|_{L^2}^2) + O\left(\sqrt{\frac{n}{Nd}}\right)$$

  - Generalization via low-degree component, interpolation via high-degree component
  - Regularization increased due to high-degree part of activation

Beyond Lazy Training

Limitation of Lazy training

- Success of deep learning depends on *learning data representation*. More complicated than random features models or variants.
- Want NNs to move moderately away from initialization.
- Random features models restricted, having trouble learning single neuron function [MBM17].
- Nevertheless, NTK may be advantageous for small-sample datasets [ADLS+19]

Mean-field perspective

- Viewing parameters as a *probability distribution* [MMN18, CB18]

$$f(\boldsymbol{x};\boldsymbol{\theta}) = \int a\sigma(\boldsymbol{w}^\top \boldsymbol{x})\, \widehat{\rho}_N(da, d\boldsymbol{w}), \qquad \text{where } \widehat{\rho}_N \text{ is empirical measure on } \mathbb{R}^{d+1}$$

- Under a nonstandard initialization scaling $var(w_k) \sim O\left(\frac{1}{n^2}\right)$, continuous-time SGD $\approx$ Gradient flow on probability measure, which is determined by a PDE.
- Advantage: capable of learning more functions
- Disadvantages: weak theory. Requires very large width (likely exponential in d), requires very large sample size (in general, exponential in d); the latter can be improved to polynomial dependence by adding noise [WLLM20]

Feature learning with GD

- Suppose the target function $f_*(\boldsymbol{x}) = g(\boldsymbol{U}\boldsymbol{x})$ where $\boldsymbol{U}$ is of size $d \times r$ with $d \gg r$. Assume g is of polynomial of degree p.
- NTK cannot learn the unknown subspace $\boldsymbol{U}$, thus requiring a much larger sample size $O(d^p)$
- Assuming non-degeneracy condition of Hessian of f_*, one-step GD on the squared loss using one-hidden-layer NN reveals information about $\boldsymbol{U}$, which results in improved sample complexity $O(d^2)$; see [DLS22]

Other approaches

- Classical tools in learning theory such as VC dimension insufficient because dimension is too large [BMM99]
- Bounding Rademacher using weight matrix norms [BFT17]
- Finding other good complexity measures by taking into account initialization [NLBLS18], algorithms, etc.

Theorem 2. *For any $h \geq 2$, $\gamma > 0$, $\delta \in (0,1)$ and $\mathbf{U}^0 \in \mathbb{R}^{h \times d}$, with probability $1-\delta$ over the choice of the training set $\mathcal{S} = \{\mathbf{x}_i\}_{i=1}^m \subset \mathbb{R}^d$, for any function $f(\mathbf{x}) = \mathbf{V}[\mathbf{U}\mathbf{x}]_+$ such that $\mathbf{V} \in \mathbb{R}^{c \times h}$ and $\mathbf{U} \in \mathbb{R}^{h \times d}$, the generalization error is bounded as follows:*

$$L_0(f) \leq \hat{L}_\gamma(f) + \tilde{O}\left(\frac{\sqrt{c}\|\mathbf{V}\|_F\left(\left\|\mathbf{U}-\mathbf{U}^0\right\|_F\|\mathbf{X}\|_F + \left\|\mathbf{U}^0\mathbf{X}\right\|_F\right)}{\gamma m} + \sqrt{\frac{h}{m}}\right)$$

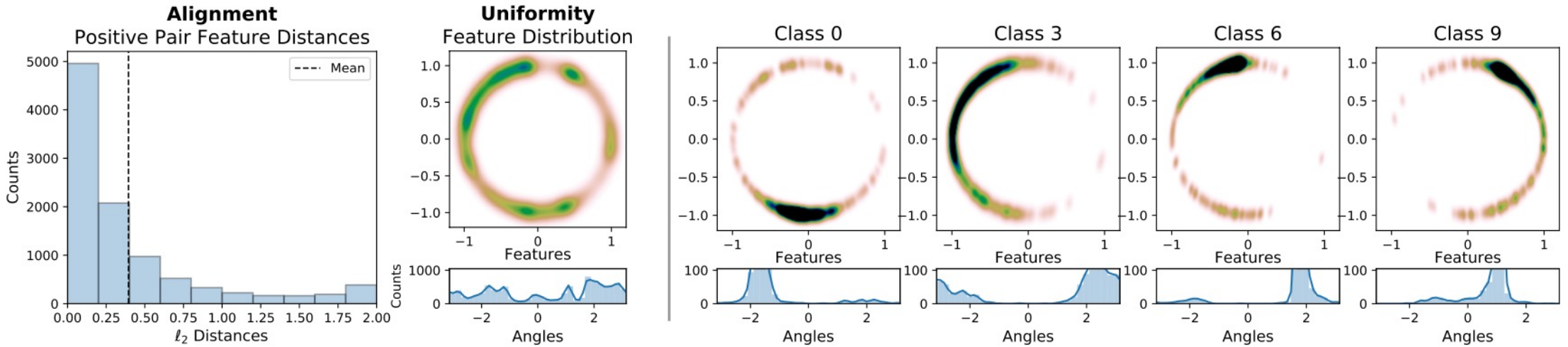
$$\leq \hat{L}_\gamma(f) + \tilde{O}\left(\frac{\sqrt{c}\|\mathbf{V}\|_F\left(\left\|\mathbf{U}-\mathbf{U}^0\right\|_F + \left\|\mathbf{U}^0\right\|_2\right)\sqrt{\frac{1}{m}\sum_{i=1}^m \|\mathbf{x}_i\|_2^2}}{\gamma\sqrt{m}} + \sqrt{\frac{h}{m}}\right).$$

Emerging phenomena, and new hope?

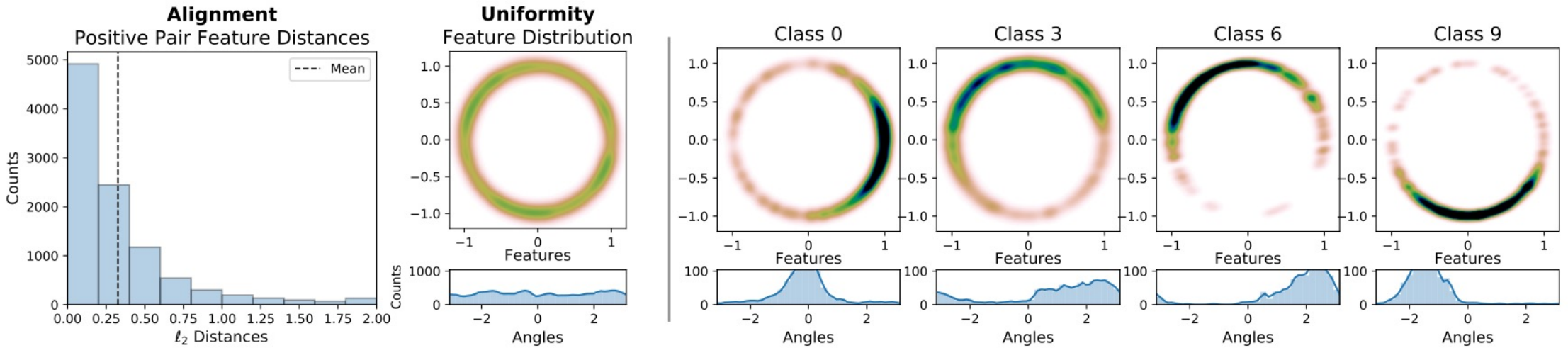
Self-supervised learning

- Representation using supervised learning $f(\boldsymbol{x}; \boldsymbol{\theta})$. Q: label intensive? How to transfer?
- With no (or very few) label information, NNs can learn good embedding, e.g., SimCLR [CKNH20]
- Clear cluster structure & meaningful learned features

- Self-supervised learning or unsupervised learning may be a bridge to understanding generalization



(b) **Supervised Predictive Learning.** Linear classification validation accuracy: 57.19%.



(c) **Unsupervised Contrastive Learning.** Linear classification validation accuracy: 28.60%.

Source: Wang, Isola, 2020

Visualizing learned features

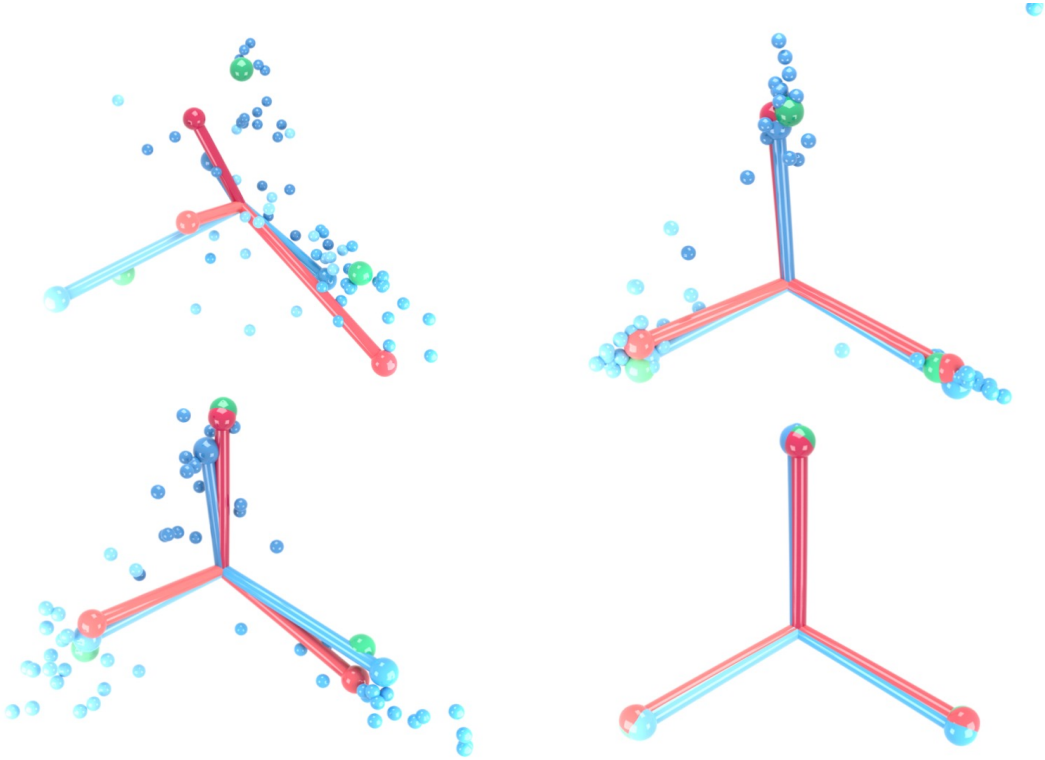
Supervised Features
(of adversarial trained Wide-ResNet)

Contrastive Features
(of adversarial-contrast trained Wide-ResNet)

Source: Wen and Li, 2021

Neural collapse

- $h = h_\theta(x)$ is last-layer activations, where $h_\theta: R^d \to R^p$, K classes
- Classifier: $Wh_\theta(x) + b$
- decomposing covariance: between-class + within-class:
- $\Sigma_T = \Sigma_B + \Sigma_W$



Source: Papyan, Han, Donoho, 2020

Clear phenomenon, clean math relations

(NC1) Variability collapse: $\Sigma_W \to 0$

(NC2) Convergence to Simplex ETF:

$$\left| \|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2 - \|\boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G\|_2 \right| \to 0 \quad \forall\, c, c'$$

$$\langle \tilde{\boldsymbol{\mu}}_c, \tilde{\boldsymbol{\mu}}_{c'} \rangle \to \frac{C}{C-1}\delta_{c,c'} - \frac{1}{C-1} \quad \forall\, c, c'.$$

(NC3) Convergence to self-duality:

$$\left\| \frac{\boldsymbol{W}^\top}{\|\boldsymbol{W}\|_F} - \frac{\dot{\boldsymbol{M}}}{\|\dot{\boldsymbol{M}}\|_F} \right\|_F \to 0$$

(NC4): Simplification to NCC:

$$\arg\max_{c'} \langle \boldsymbol{w}_{c'}, \boldsymbol{h} \rangle + b_{c'} \to \arg\min_{c'} \|\boldsymbol{h} - \boldsymbol{\mu}_{c'}\|_2$$

Intermediate layers for generalization theory?

- How about intermediate layers? Do we have neural collapse?
- Empirical work by [GGB20] demonstrates existence of *effective depth,* which is a threshold L ---below layer L within-class variances decrease but no collapse, above layer L there is neural collapse
- Can we decompose overparametrized deep NNs into “representation learning component” and “interpolation component”? If so, helpful for generalization & transfer learning

Thank you!

Contact: yiqiao.zhong@wisc.edu
Office: MSC 1122