NEURAL INFORMATION PROCESSING SYSTEMS

From Tempered to Benign Overfitting in ReLU Neural Networks

Guy Kornowski*   Gilad Yehudai*   Ohad Shamir

Weizmann Institute of Science

Spotlight presentation

WEIZMANN INSTITUTE OF SCIENCE

*Equal contribution

Overfitting puzzle

Overfitting puzzle

Main Question: Why do overparameterized neural networks generalize?

Overfitting puzzle

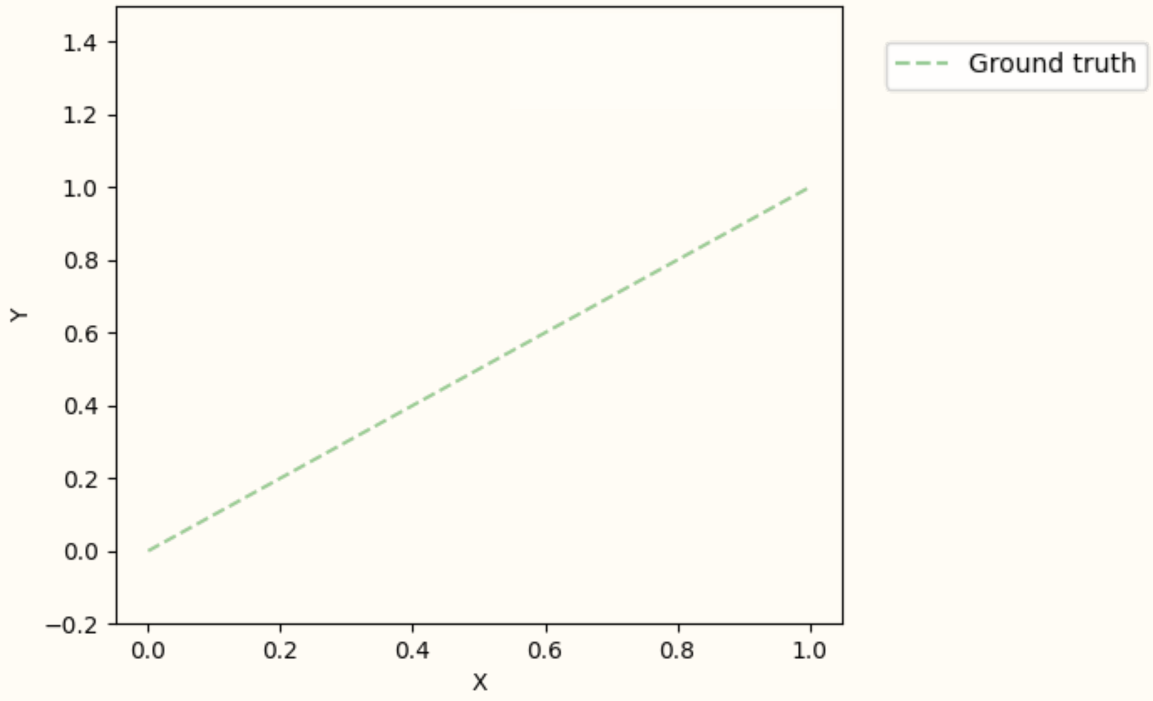
Main Question: Why do overparameterized neural networks generalize?

- Even when trained to fit noisy samples, even without regularization…

Overfitting puzzle

Main Question: Why do overparameterized neural networks generalize?

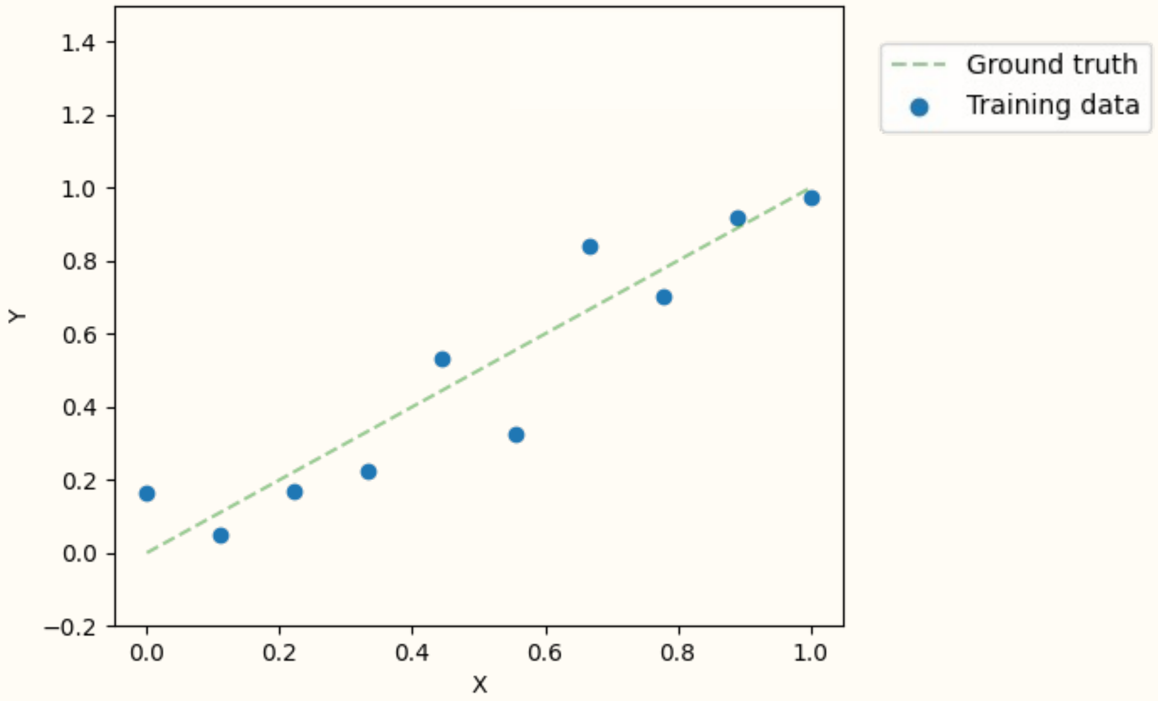
- Even when trained to fit noisy samples, even without regularization…



Overfitting puzzle

Main Question: Why do overparameterized neural networks generalize?

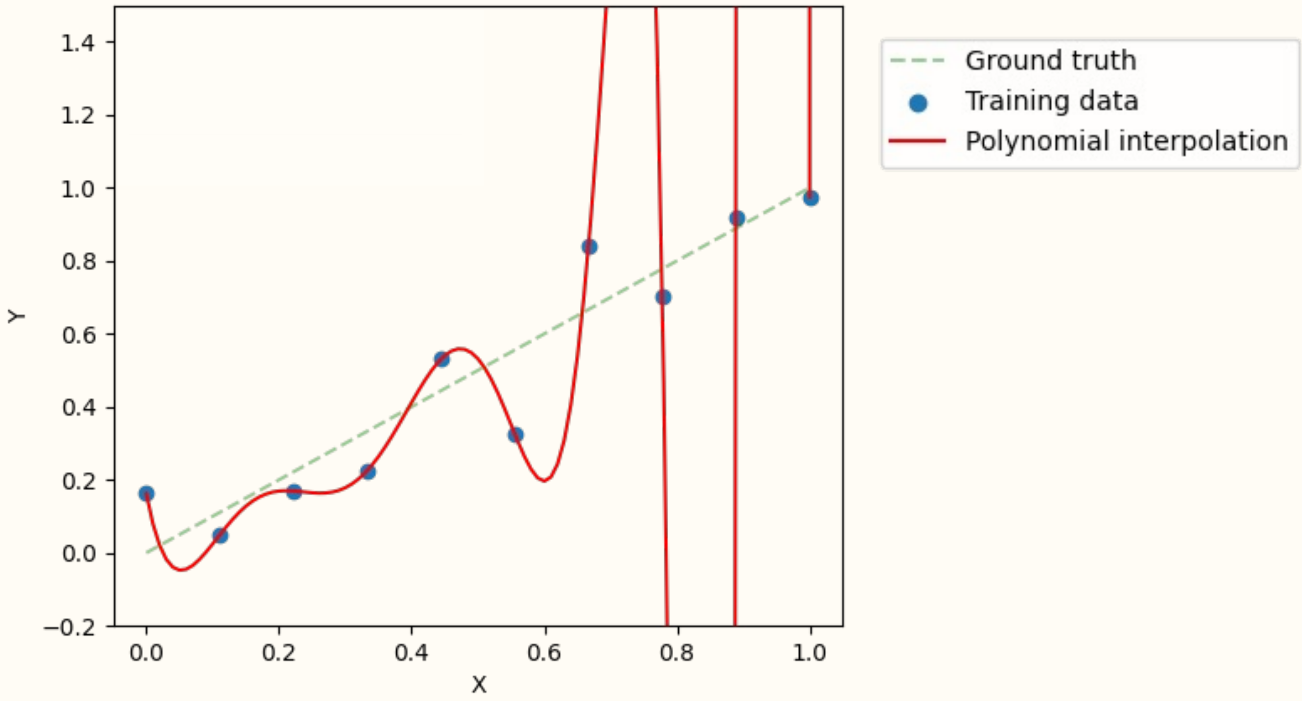
- Even when trained to fit noisy samples, even without regularization…



Overfitting puzzle

Main Question: Why do overparameterized neural networks generalize?

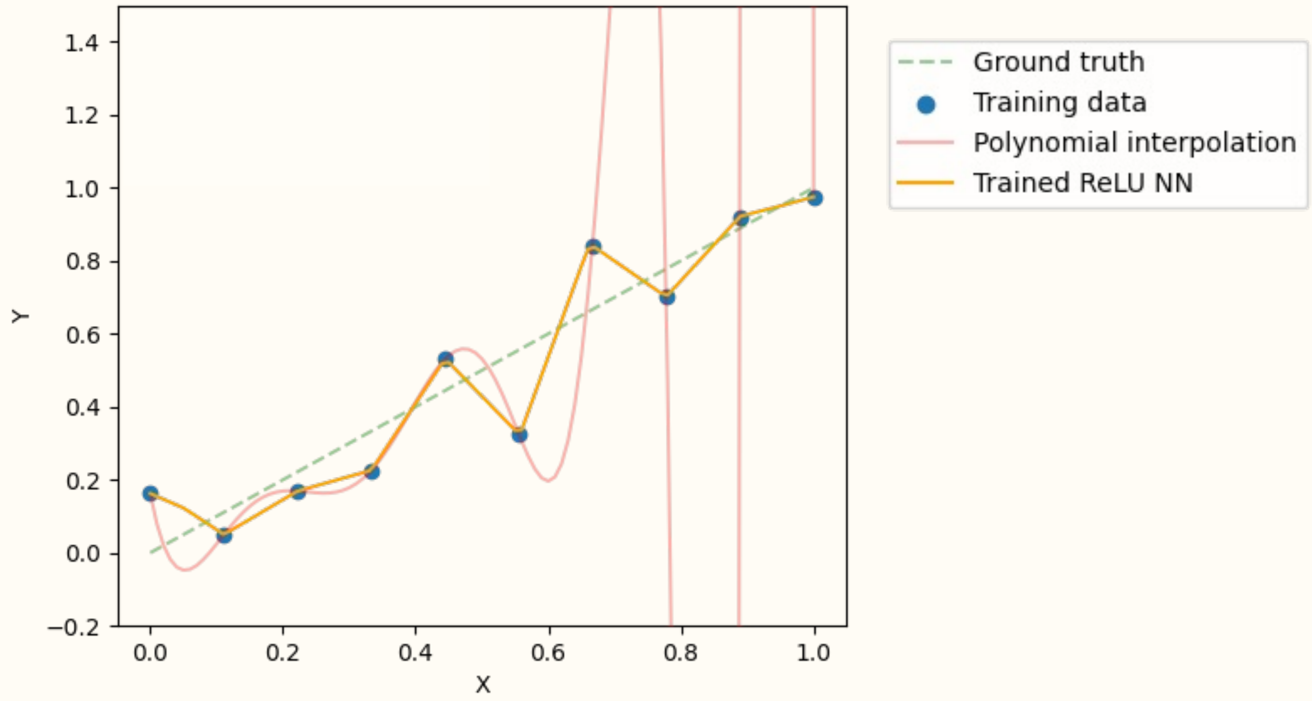
- Even when trained to fit noisy samples, even without regularization…



Overfitting puzzle

Main Question: Why do overparameterized neural networks generalize?

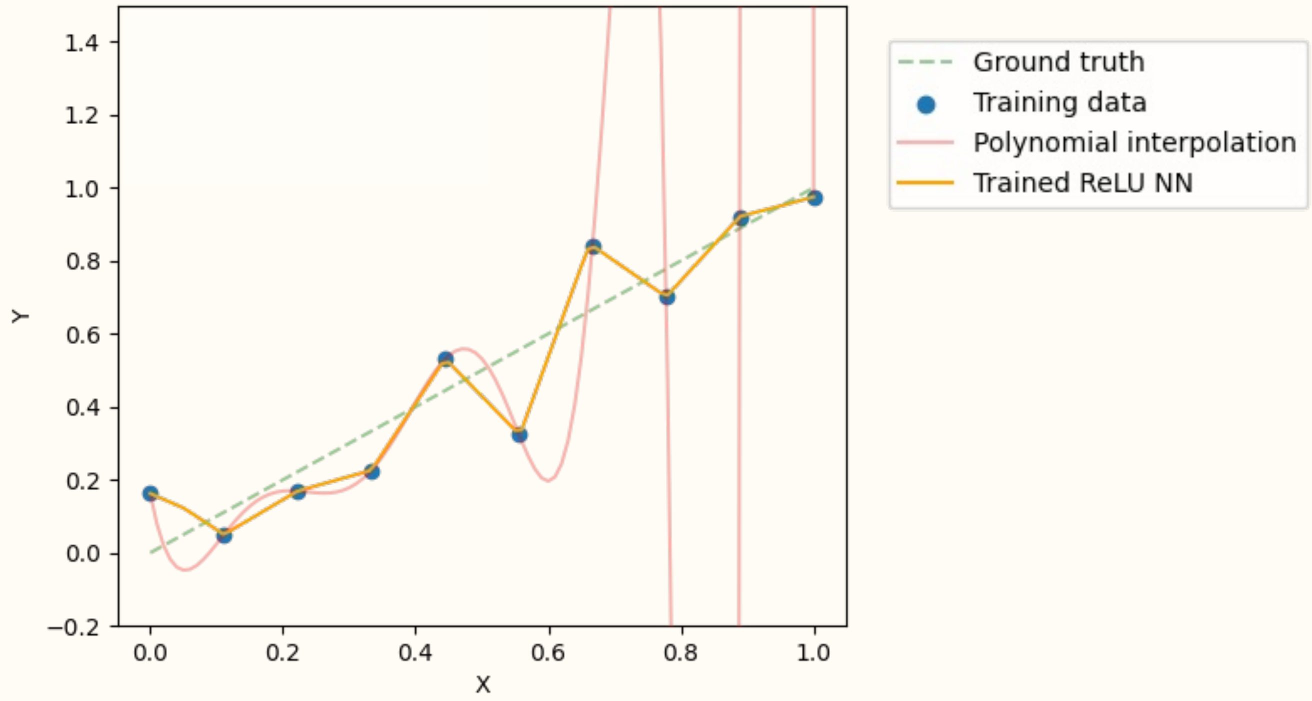
- Even when trained to fit noisy samples, even without regularization…



Overfitting puzzle

Main Question: Why do overparameterized neural networks generalize?

- Even when trained to fit noisy samples, even without regularization…



- Seems to defy classical learning theory, “Occam’s razor”…

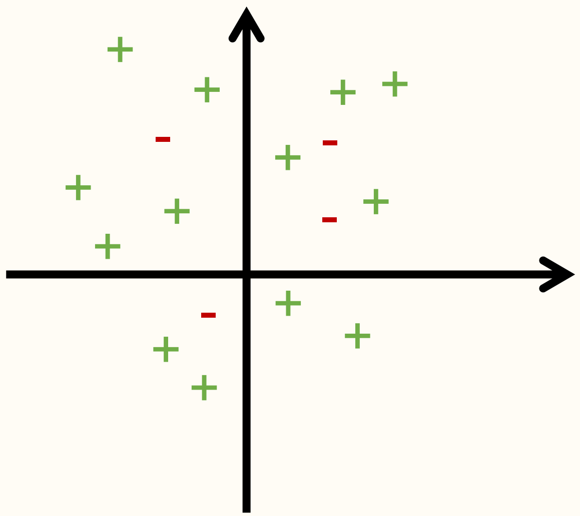
Setting

Setting

- Noisy "classification" data: $S = (x_i, y_i)_{i=1}^{m} \subset \mathbb{R}^d \times \{\pm 1\}$

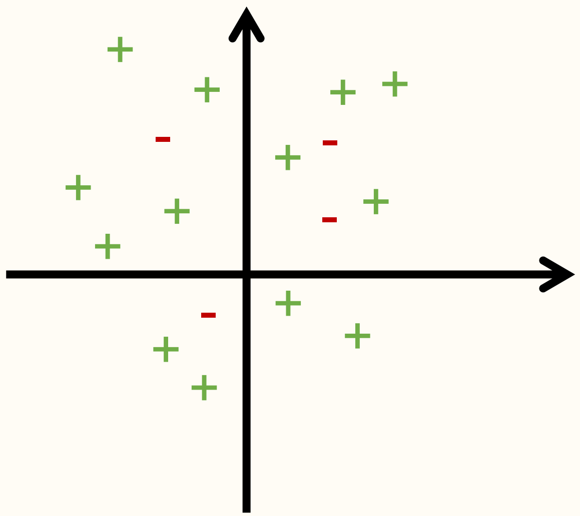
Setting

- Noisy "classification" data: $S = (x_i, y_i)_{i=1}^m \subset \mathbb{R}^d \times \{\pm 1\}$
  - Ground truth $f^*(x_i) \equiv 1$, each y_i flipped w.p. $p \in \left[0, \frac{1}{2}\right)$



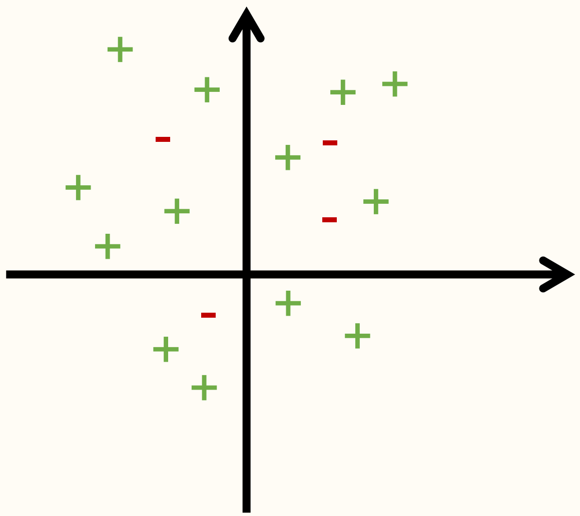
Setting

- Noisy "classification" data: $S = (x_i, y_i)_{i=1}^m \subset \mathbb{R}^d \times \{\pm 1\}$
  - Ground truth $f^*(x_i) \equiv 1$, each y_i flipped w.p. $p \in \left[0, \frac{1}{2}\right)$
- Two-layer ReLU neural network $N_\theta(x) \coloneqq \sum_{j=1}^n a_j \cdot \left[w_j \cdot x + b_j\right]_+$

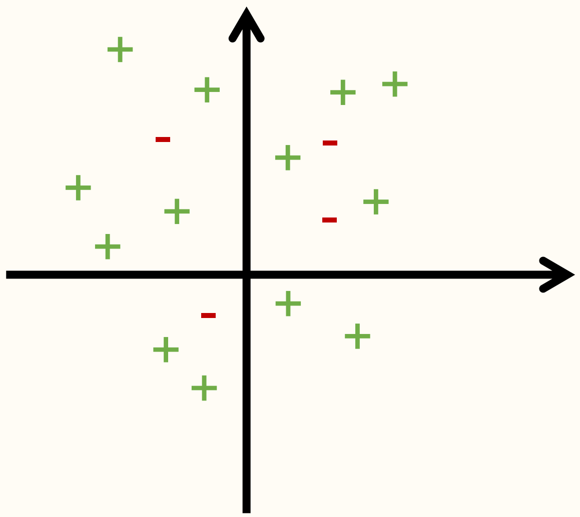


Setting

- Noisy "classification" data: $S = (x_i, y_i)_{i=1}^{m} \subset \mathbb{R}^d \times \{\pm 1\}$
  - Ground truth $f^*(x_i) \equiv 1$, each y_i flipped w.p. $p \in \left[0, \frac{1}{2}\right)$
- Two-layer ReLU neural network $N_\theta(x) \coloneqq \sum_{j=1}^{n} a_j \cdot \left[w_j \cdot x + b_j\right]_+$
- Network interpolates dataset: $y_i N_\theta(x_i) > 0, \ \forall i \in [m]$

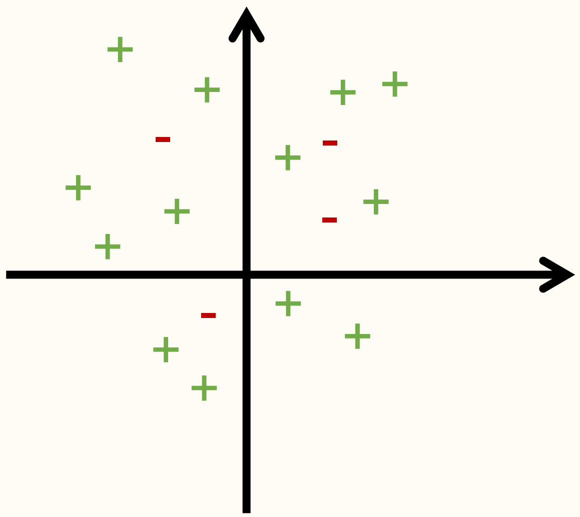


Types of overfitting (following Mallinar et al. ’22)



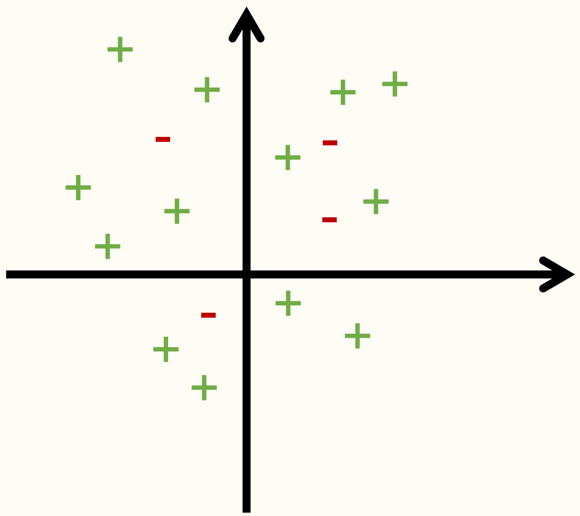
Types of overfitting (following Mallinar et al. '22)

- Analyze the *clean* test error $L(N_\theta) \coloneqq \Pr_x[N_\theta(x) \leq 0]$



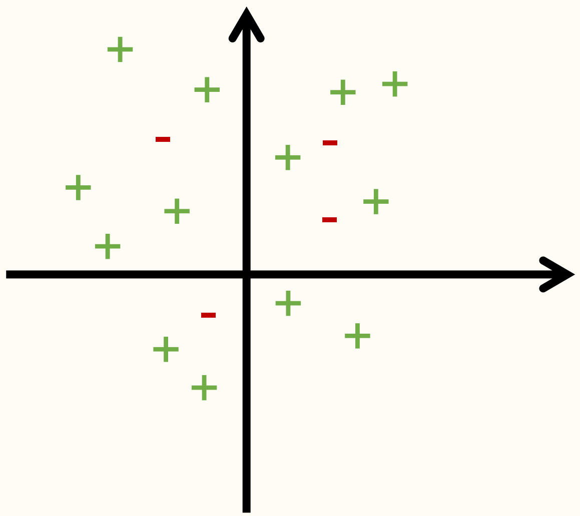
Types of overfitting (following Mallinar et al. '22)

- Analyze the *clean* test error $L(N_\theta) \coloneqq \Pr_x[N_\theta(x) \leq 0]$
  - The overfitting is called "benign" if $L(N_\theta) \to 0$ [Bartlett et al. '20]



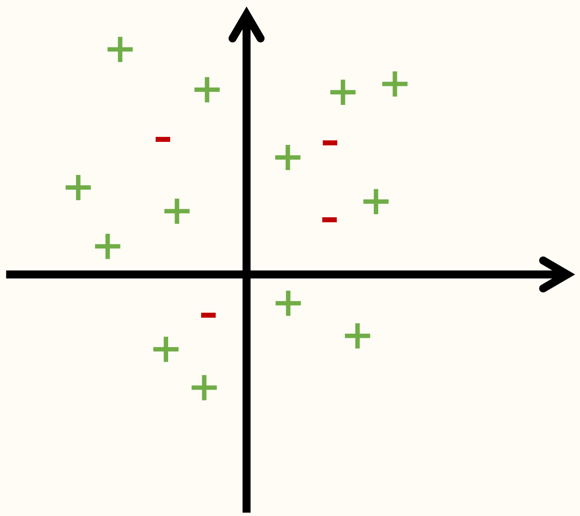
Types of overfitting (following Mallinar et al. '22)

- Analyze the *clean* test error $L(N_\theta) \coloneqq \Pr_x[N_\theta(x) \leq 0]$
  - The overfitting is called "benign" if $L(N_\theta) \to 0$ [Bartlett et al. '20]
  - The overfitting is called "tempered" if $L(N_\theta) \in \left(0, \frac{1}{2}\right)$



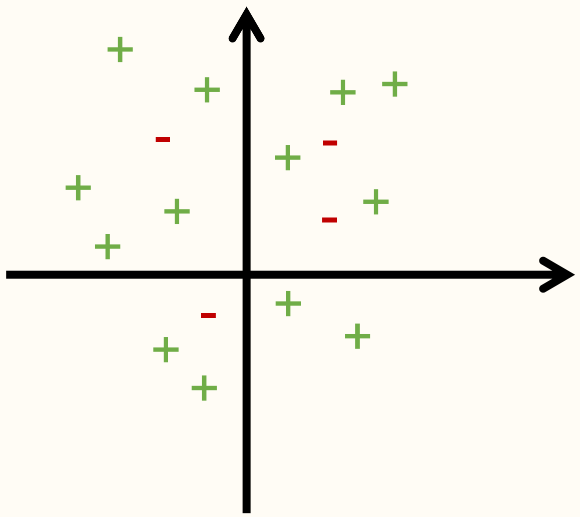
Types of overfitting (following Mallinar et al. '22)

- Analyze the *clean* test error $L(N_\theta) \coloneqq \Pr_x[N_\theta(x) \leq 0]$
  - The overfitting is called "benign" if $L(N_\theta) \rightarrow 0$ [Bartlett et al. '20]
  - The overfitting is called "tempered" if $L(N_\theta) \in \left(0, \frac{1}{2}\right)$
    - ☆ *Special case of ineterest is when $L(N_\theta)$ scales with p, e.g. $L(N_\theta) \approx p$*



Types of overfitting (following Mallinar et al. ’22)

- Analyze the *clean* test error $L(N_\theta) \coloneqq \Pr_x[N_\theta(x) \leq 0]$
  - The overfitting is called “benign” if $L(N_\theta) \to 0$ [Bartlett et al. ’20]
  - The overfitting is called “tempered” if $L(N_\theta) \in \left(0, \frac{1}{2}\right)$
    - ☆ *Special case of ineterest is when $L(N_\theta)$ scales with p, e.g. $L(N_\theta) \approx p$*
  - The overfitting is called “catastrophic” if $L(N_\theta) \to \frac{1}{2}$



Main technical tool: Implicit bias

Gradient based training with certain losses (e.g. logistic) drives θ towards a KKT point of the margin maximization problem

$$\min \|\theta\|^2 \quad s.t \quad y_i N_\theta(x_i) \geq 1 \;\; \forall i \in [m]$$

[Lyu & Li ‘20, Ji & Telgarsky ‘20]

From tempered to benign overfitting

From tempered to benign overfitting

Theorem: In dimension $d = 1$, with noise level p, w.h.p. over the sample any KKT point θ satisfies $L(N_\theta) \in \left(p^5, \sqrt{p}\right)$.

Moreover, any local minimum of max margin θ satisfies $L(N_\theta) \approx p$.

From tempered to benign overfitting

Theorem: In dimension $d = 1$, with noise level p, w.h.p. over the sample any KKT point θ satisfies $L(N_\theta) \in \left(p^5, \sqrt{p}\right)$.

Moreover, any local minimum of max margin θ satisfies $L(N_\theta) \approx p$.

Theorem: In dimension $d \gtrsim \mathrm{poly}(m) \log(1/\epsilon)$, under some assumptions*, w.h.p. over the sample, KKT points θ satisfy $L(N_\theta) \leq \epsilon$.

Equivalently, $L(N_\theta) \lesssim \exp(-d)$.

*We give different sets of assumption, probably not minimal

From tempered to benign overfitting

Theorem: In dimension $d = 1$, with noise level p, w.h.p. over the sample any KKT point θ satisfies $L(N_\theta) \in \left(p^5, \sqrt{p}\right)$.

Moreover, any local minimum of max margin θ satisfies $L(N_\theta) \approx p$.

Theorem: In dimension $d \gtrsim \mathrm{poly}(m) \log(1/\epsilon)$, under some assumptions*, w.h.p. over the sample, KKT points θ satisfy $L(N_\theta) \leq \epsilon$.

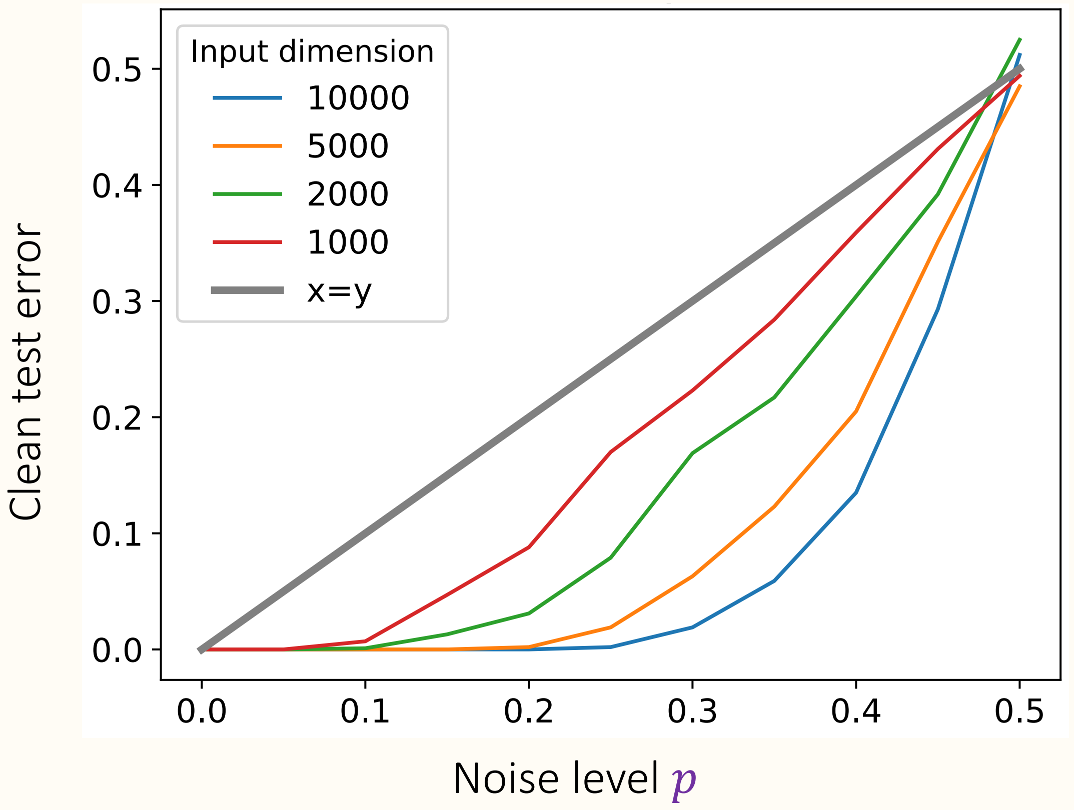
Equivalently, $L(N_\theta) \lesssim \exp(-d)$.

More results and details in paper…

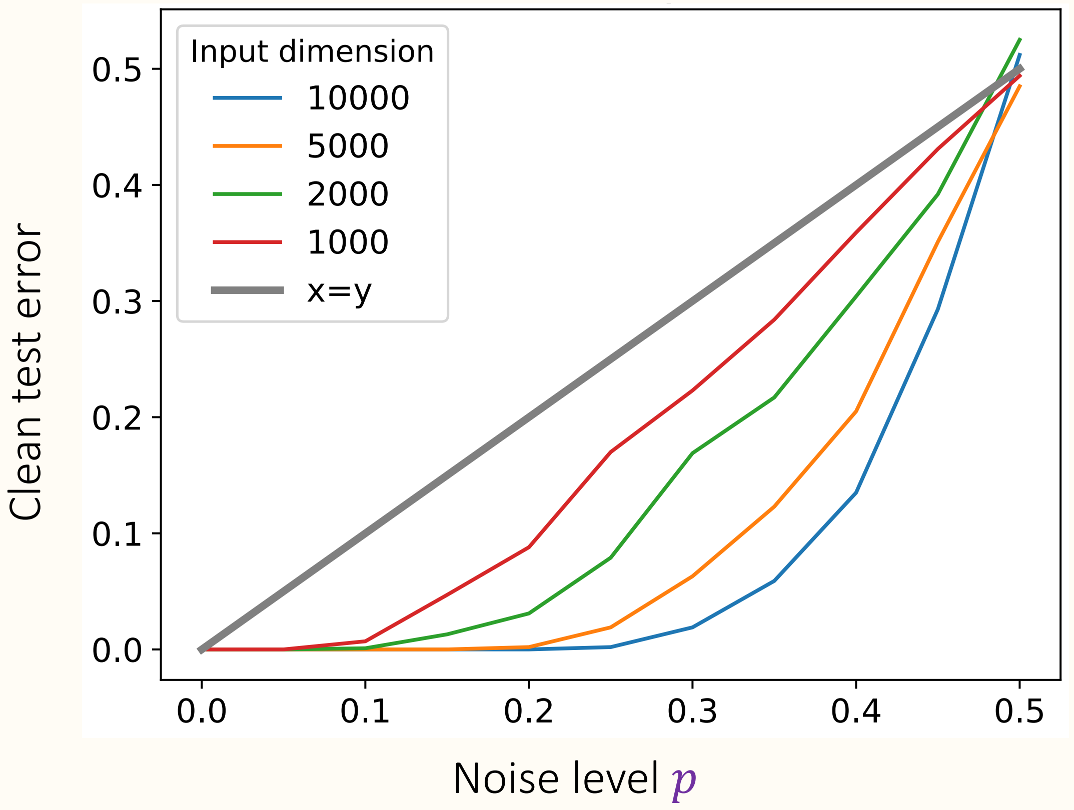
*We give different sets of assumption, probably not minimal

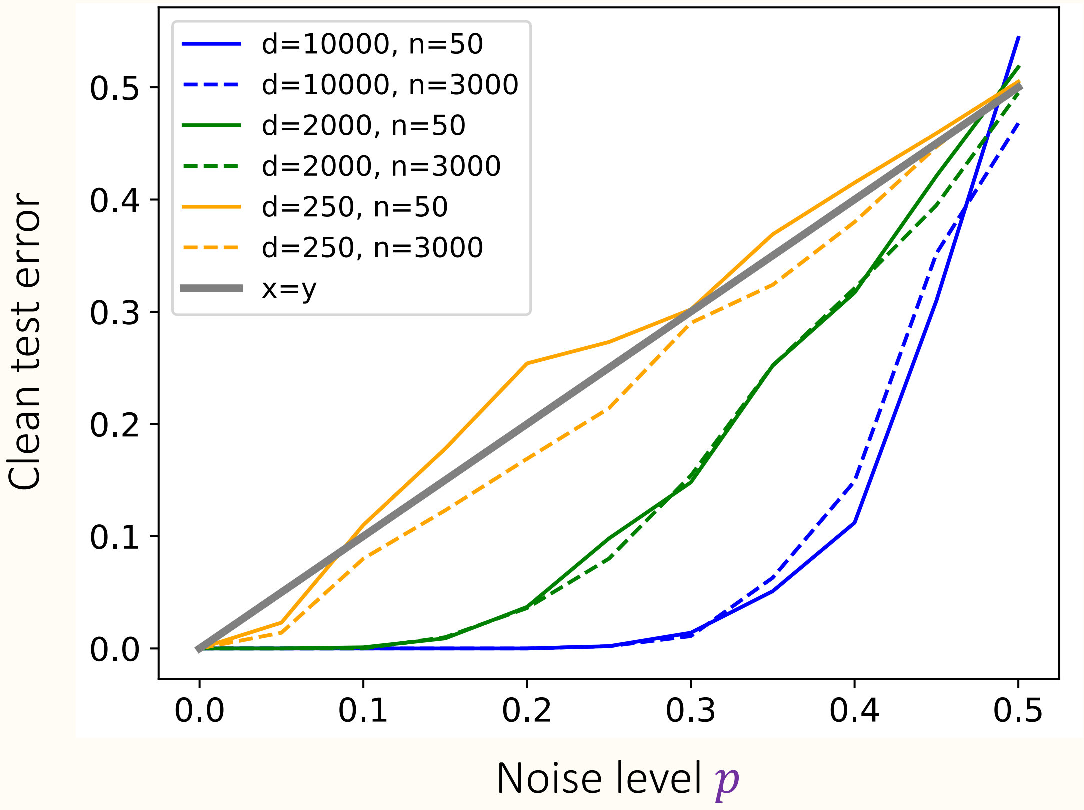
Empirical study of intermediate dimensions

Empirical study of intermediate dimensions



Empirical study of intermediate dimensions





Thanks!