Generalization Theory for Deep Learning

Basic version: finite hypothesis class

Finite hypothesis class: with probability $1-\delta$ over the choice of a training set of size n, for a bounded loss ℓ, we have

$$\sup_{f\in\mathcal{F}}\left|\frac{1}{n}\sum_{i=1}^{n}\ell(f(x_i),y_i)-\mathbb{E}_{(x,y)\sim D}\left[\ell(f(x),y)\right]\right| = O\left(\sqrt{\frac{\log|\mathcal{F}|+\log 1/\delta}{n}}\right)$$

Pf: for a fixed f, by Hoeffding's inequality

w.p. $1-\frac{\delta}{|\mathcal{F}|}$, $\text{gen\_error}(f) = O\left(\sqrt{\frac{\log\left(\frac{|\mathcal{F}|}{\delta}\right)}{n}}\right)$

Union bound: event$_1$, ... event$_m$

$$P\left(\bigcup_{i=1}^{m}\text{event}_i\right) \le \sum_{i=1}^{n} P(\text{event}_i)$$

choose event$_f$: $\text{gen\_error}(f) \not> \sqrt{\frac{\log\left(\frac{|\mathcal{F}|}{\delta}\right)}{n}}$

$$P\left(\bigcup_{f\in\mathcal{F}}\text{event}_f\right) \le \sum_{f\in\mathcal{F}} P(\text{event}_f) = |\mathcal{F}|\cdot\frac{\delta}{|\mathcal{F}|} = \delta$$

$\Rightarrow \forall f,\ P\left(\text{gen\_error}(f) < \sqrt{\frac{\log(|\mathcal{F}|/\delta)}{n}}\right) \not< 1-\delta$ □

VC-Dimension

Motivation: Do we need to consider **every** classifier in $\mathcal{F}$?
Intuitively, **pattern of classifications** on the training set should suffice. (Two predictors that predict identically on the training set should generalize similarly).

Let $\mathcal{F} = \{f : \mathbb{R}^d \to \{+1, -1\}\}$ be a class of binary classifiers.

The **growth function** $\Pi_{\mathcal{F}} : \mathbb{N} \to \mathbb{F}$ is defined as:

$$\Pi_{\mathcal{F}}(m) = \max_{(x_1, x_2, \ldots, x_m)} \left| \left\{ (f(x_1), f(x_2), \ldots, f(x_m)) \mid f \in \mathcal{F} \right\} \right|.$$

max: 2^m

The VC dimension of $\mathcal{F}$ is defined as:

$$\text{VCdim}(\mathcal{F}) = \max\{m : \Pi_{\mathcal{F}}(m) = 2^m\}.$$

VC-dimension Generalization bound

Theorem (Vapnik-Chervonenkis): with probability $1-\delta$ over the choice of a training set, for a bounded loss ℓ, we have

$$\sup_{f\in\mathcal{F}}\left|\frac{1}{n}\sum_{i=1}^{n}\ell(f(x_i),y_i)-\mathbb{E}_{(x,y)\sim D}\left[\ell(f(x),y)\right]\right| = O\left(\sqrt{\frac{\mathrm{VCdim}(\mathcal{F})\log n+\log 1/\delta}{n}}\right)$$

Examples:

- Linear functions: VC-dim = O(dimension)
- Neural network: VC-dimension of fully-connected net with width W and H layers is $\widetilde{\Theta}(WH)$ (Bartlett et al., '17).

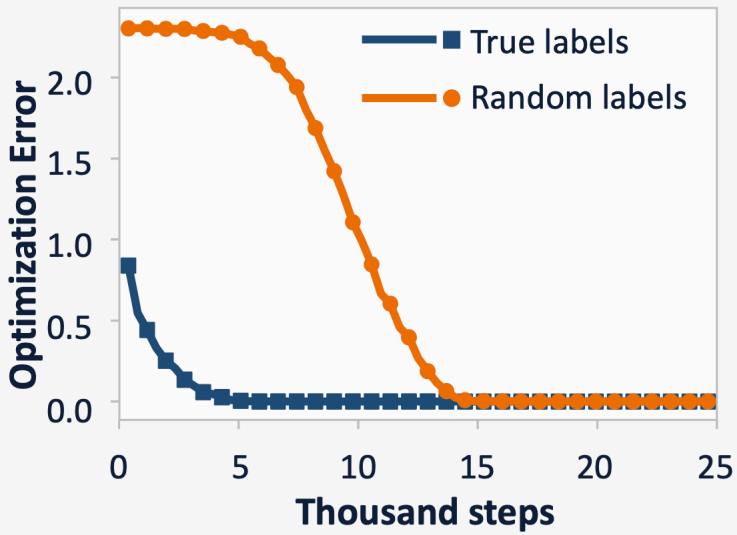
Problems with VC-dimension bound

if $n <$ #params

1. In over-parameterized regime, bound >> 1.
2. Cannot explain the random noise phenomenon:
   - Neural networks that fit random labels and that fit true labels have the same VC-dimension.

Practice: gradient descent

$$\theta(t+1) \leftarrow \theta(t) - \eta \frac{\partial L(\theta(t))}{\partial \theta(t)}$$



Optimization error -> **0** for both ***true labels*** and ***random labels*** !

Zhang Bengio Hardt Recht Vinyals 2017
Understanding DL Requires Rethinking Generalization

PAC Bayesian Generalization Bounds

Setup: Let P be a prior over function in class $\mathcal{F}$, let Q be the posterior (after algorithm's training).

Theorem: with probability $1-\delta$ over the choice of a training set, for a bounded loss ℓ, we have

$$\sup_{f\in\mathcal{F}} \left| \frac{1}{n}\sum_{i=1}^{n} \ell(f(x_i), y_i) - \mathbb{E}_{(x,y)\sim D}\left[\ell(f(x), y)\right] \right| = O\left(\sqrt{\frac{KL(Q \,||\, P) + \log 1/\delta}{n}}\right)$$

prior: NN's parameter gaussian

Q : Gaussian

Rademacher Complexity

Intuition: how well can a classifier class **fit random noise?**

(Empirical) **Rademacher complexity:** For a training set $S = \{x_1, x_2, \ldots, x_n\}$, and a class $\mathcal{F}$, denote:

$$\hat{R}_n(S) = \mathbb{E}_\sigma \sup_{f \in \mathcal{F}} \sum_{i=1}^{n} \sigma_i f(x_i) \, .$$

where $\sigma_i \sim \text{Unif}\{+1, -1\}$ (Rademacher R.V.).

(Population) **Rademacher complexity:**

$$R_n = \mathbb{E}_S \left[\hat{R}_n(s) \right].$$

Rademacher Complexity Generalization Bound

Theorem: with probability $1 - \delta$ over the choice of a training set, for a bounded loss ℓ, we have

$$\sup_{f \in \mathcal{F}} \left| \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i) - \mathbb{E}_{(x,y)\sim D} \left[\ell(f(x), y) \right] \right| = O\left(\frac{\hat{R}_n}{n} + \sqrt{\frac{\log 1/\delta}{n}} \right)$$

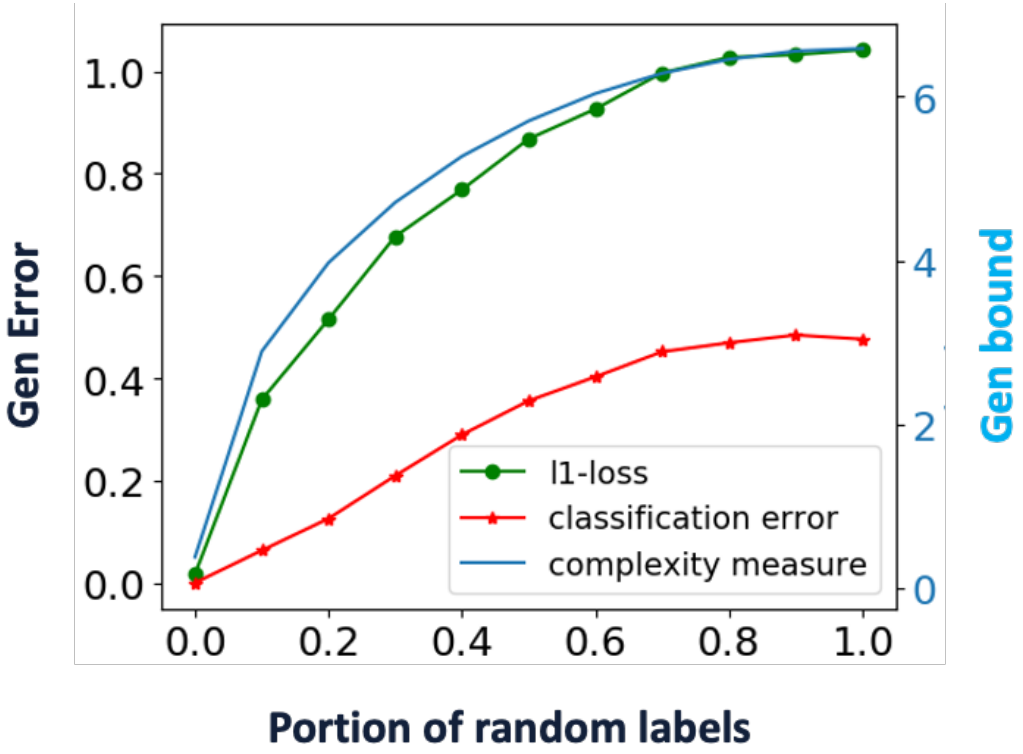
$\hat{R}_n \asymp O(\sqrt{\eta})$

and

$$\sup_{f \in \mathcal{F}} \left| \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i) - \mathbb{E}_{(x,y)\sim D} \left[\ell(f(x), y) \right] \right| = O\left(\frac{R_n}{n} + \sqrt{\frac{\log 1/\delta}{n}} \right)$$

Kernel generalization bound

Use Rademacher complexity theory, we can obtain a generalization bound $O(\sqrt{y^\top (H^*)^{-1} y/n})$ where $y \in \mathbb{R}^n$ are n labels, and $H^* \in \mathbb{R}^{n \times n}$ is the kernel (e.g., NTK) matrix.



Norm-based Rademacher complexity bound

Theorem: If the activation function is σ is ρ-Lipschitz. Let

$$\mathcal{F} = \{x \mapsto W_{H+1}\sigma(W_h\sigma(\cdots\sigma(W_1x)\cdots), \|W_h^T\|_{1,\infty} \leq B \forall h \in [H]\}$$

then $R_n(\mathcal{S}) \leq \|X^\top\|_{2,\infty}(2\rho B)^{H+1}\sqrt{2\ln d}$ where

$X = [x_1, \ldots, x_n] \in \mathbb{R}^{d\times n}$ is the input data matrix.

Comments on generalization bounds

- When plugged in real values, the bounds are rarely non-trivial (i.e., smaller than 1)
- "*Fantastic Generalization Measures and Where to Find them*" by Jiang et al. '19 : large-scale investigation of the correlation of extant generalization measures with true generalization.

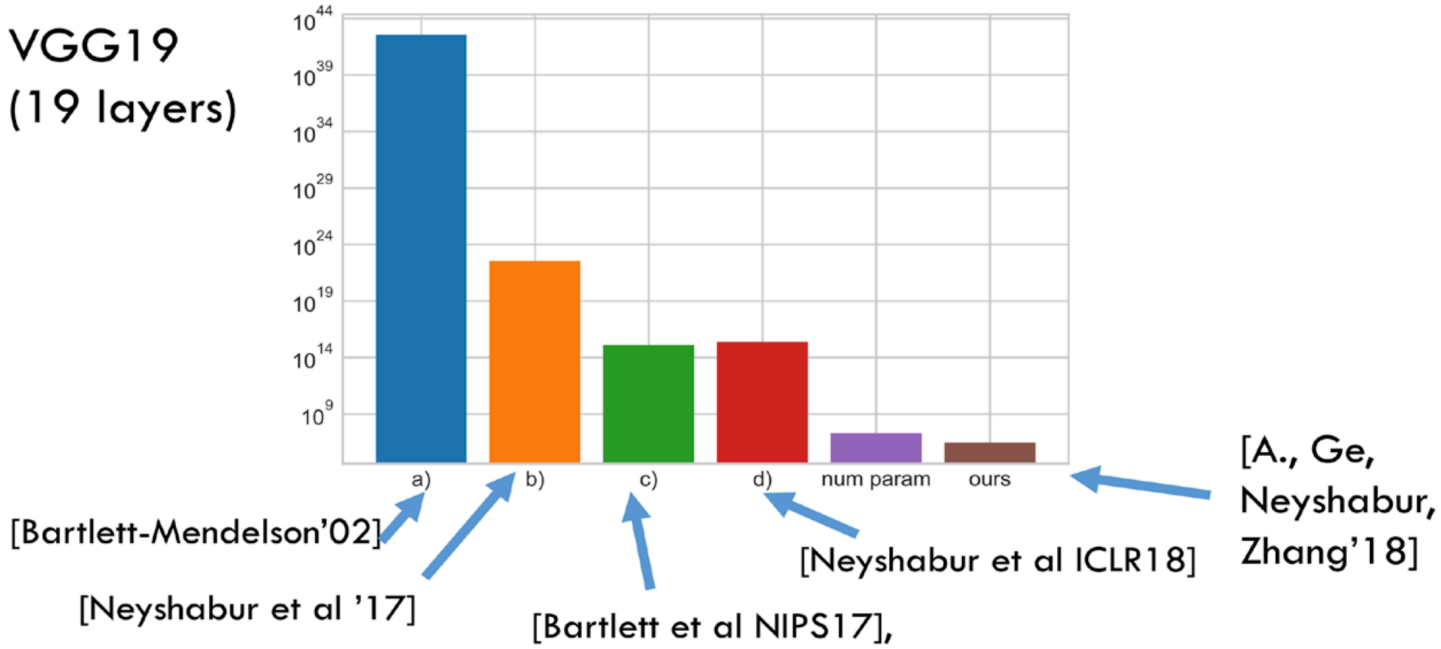


Image credits to Andrej Risteski

Comments on generalization bounds

- Uniform convergence may be unable to explain generalization of deep learning [Nagarajan and Kolter, '19]
  - Uniform convergence: a bound for all $f \in \mathcal{F}$
  - Exists example that 1) can generalize, 2) uniform convergence fails.

bias^2 + var

- Rates:
  - Most bounds: $1/\sqrt{n}$.
  - Local Rademacher complexity: $1/n$.

Separation between NN and kernel

- For approximation and optimization, neural network has no advantage over kernel. Why NN gives better performance: generalization.

- [Allen-Zhu and Li '20] Construct a class of functions $\mathcal{F}$ such that $y = f(x)$ for some $f \in \mathcal{F}$:
  - no kernel is sample-efficient;
  - Exists a neural network that is sample-efficient.

need exp(d) samples

poly(d) samples

Separation between NN and kernel

Def: Kernel method is linear method with an embedding $\phi: \mathbb{R}^d \to \mathcal{H}$, Hilbert space

$\Rightarrow$ it turns an element $f \in \mathcal{H}$ into a prediction function $g = \langle f, \phi(x) \rangle \triangleq f(x)$

The method uses samples $\{x_i\}_{i=1}^n$, $x_i \in \mathbb{R}^d$

observes $\{y_i\}_{i=1}^n$

$f \in \text{span}(\{\phi(x_i)\}_{i=1}^n)$

Example: $\underset{f}{\text{argmin}} \frac{1}{n} \sum_{i=1}^n (y_i - \langle f, \phi(x_i) \rangle)^2 + \lambda \|f\|_{\mathcal{H}}^2$

$\Rightarrow f \in \text{span}(\{\phi(x_i)\})$

Separation between NN and kernel

Thm ∃ a class of functions $C \subseteq \{\mathbb{R}^d \to \mathbb{R}\}$ and a distribution over $\mathbb{R}^d$ s.t.

i) ∀ kernel method, if it satisfies that ∀ $c \in C$, given $y_i = c(x_i)$ if $\mathbb{E}_u\left[(c(x) - \langle f, \psi(x)\rangle)^2\right] \leq \frac{1}{q}$ they you need $n \geq 2^{d-1}$

ii) ∃ simple procedure (not kernel) that you can output correct c, using $n = d$ samples

★ NN can simulate this procedure

Separation between NN and kernel

Pf: $\mathcal{U}$: uniform distribution over $\{-1, 1\}^d$

$$\mathcal{C} = \left\{ C_S = \prod_{s \in S} x_i, \; S \subset \{1, \cdots, d\} \right\}$$

Pf of part iii), choose $\underbrace{\begin{pmatrix} -1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}}_{e_1}, \begin{pmatrix} 1 \\ -1 \\ \vdots \\ 1 \end{pmatrix}, \cdots, \underbrace{\begin{pmatrix} 1 \\ 1 \\ \vdots \\ -1 \end{pmatrix}}_{e_d}$

$\Rightarrow y_i = C_S(e_i)$, if $i \in S$, $C_S(e_i) = -1$

if $i \notin S$, $C_S(e_i) = 1$

$\Rightarrow$ identify which i is in S

Separation between NN and kernel

part i) C is a basis for $\{f: \{-1,1\}^d \to \mathbb{R}\}$ w.r.t. $\mathcal{U}$ (by symmetry)

$C_S, C_{S'}$ $\mathbb{E}_{x\sim\mathcal{U}}[C_S(x)\cdot C_{S'}(x)] = \begin{cases} 0 & \text{if } S \neq S' \\ 1 & \text{if } S = S' \end{cases}$

$\Rightarrow \forall f, \exists \{a_S\}, f = \sum a_S \cdot C_S$

Goal: $\mathbb{E}_{x\sim\mathcal{U}}\left[\left(C_S^{*}(x) - \langle f, \phi(x)\rangle\right)^2\right]$

Since f is in span $\{\phi(x_i)\}$

$f = \sum_{i=1}^{n} a_i \phi(x_i)$, $f(x) = \sum_{i=1}^{n} a_i \langle \phi(x_i), \phi(x)\rangle$

$x \mapsto \langle \phi(x_i), \phi(x)\rangle$

$= \sum_{S\subseteq[d]} \lambda_{i,S} C_S(x)$

Separation between NN and kernel

$$\mathbb{E}_{x\sim\mu}\left[\left(C_{f^*}(x) - \langle f, \phi(x)\rangle\right)^2\right] \qquad (*)$$

$$= \mathbb{E}_{x\sim\mu}\left[\left(C_{f^*}(x) - \sum_{S\subseteq[d]}\sum_{i=1}^{n} a_i \lambda_{i,S}\, C_S(x)\right)^2\right]$$

$$= \left(1 - \sum_{i=1}^{n} a_i \lambda_{i,S^*}\right)^2 + \sum_{S\neq S^*}\left(\sum_{i=1}^{n} a_i \lambda_{i,S}\right)^2$$

by assumption $(*) \le \frac{1}{4}$

$$\Rightarrow \left(1 - \sum_{i=1}^{n} a_i \lambda_{i,S^*}\right)^2 \le \frac{1}{4}$$

$$\sum_{S\neq S^*}\left(\sum_{i=1}^{n} a_i \lambda_{i,S}\right)^2 \le \frac{1}{4}$$

Separation between NN and kernel

Notations:

$\Lambda : 2^d \times 4$

$\Lambda_{S,i} = \lambda_{i,S}$

$A = n \times 2^d$

$A_{i,S^*} = a_{i,S^*}$

$\Omega = \Lambda A : 2^d \times 2^d$ of rank at most n

$\Omega_{S^*,S} = \sum_{i=1}^{n} a_{i,S^*} \cdot \lambda_{i,S^*}$

$\Rightarrow \Omega_{S^*,S} \geq \frac{2}{3}$

$\sum_{S \neq S^*} \Omega^2_{S,S^*} \leq \frac{1}{4}$

Separation between NN and kernel

$\Omega = \mathrm{diag}(\Omega) + \Omega'$, Ω' : off-diagonal

$\|\Omega'\|_F^2 = \sum \mathrm{eigen}^2(\Omega') \le \frac{2^d}{q}$

$\Rightarrow$ Ω' has at most $\frac{2^d}{c}$ eigenvalues $\ge \frac{2}{3}$

consider subspace with eigenvalue $< \frac{2}{3}$

$\longrightarrow$ dim $\ge \frac{3}{4} \cdot 2^d$

$\forall x \in$ this subspace

$\|\Omega x\|_2 = \|\mathrm{diag}(\Omega)x + \Omega' x\|_2$

$\ge \|\mathrm{diag}(\Omega)x\|_2 - \|\Omega' x\|_2$

$\frac{2}{3}\|x\|_2 - \frac{2}{3}\|x\|_2 > 0$

$\mathrm{rank}(\Omega) \ge \frac{3}{4} \cdot 2^d$, $n \ge \frac{3}{4} \cdot 2^d \ge 2^{d-1}$