Mathematics of Deep Learning
Lecture 7: Over-parametrization, Convergence, Neural Tangent Kernel, Generalization

Prof. Predrag R. Jelenković

Time: Tuesday 4:10-6:40pm

Dept. of Electrical Engineering
Columbia University , NY 10027, USA
Office: 812 Schapiro Research Bldg.
Phone: (212) 854-8174
Email: predrag@ee.columbia.edu
URL: http://www.ee.columbia.edu/∼predrag

Over-Parametrization

- **Over-parametrization**
  - Mathematically speaking: Passes the width $m \to \infty$ (this is the opposite of the previous work on expressiveness, where the goal was to minimize the $\#$ of neurons)
  - Mathematical tools: Laws of large numbers
    - "Smooth out" the layer functions
  - In large width limit NNs behave very much like linear models, even though they are nonlinear in general.
    - Connect NNs to Kernels: Today we'll talk about: Neural Tangent Kernel: Convergence and Generalization in Neural Networks, by A. Jacot, F. Gabriel and C. Hongler, NIPS 2018.
    - Shaw that all local minima are global
    - Shaw convergence to global minima
    - Shaw generalization bounds

No Bad Local Minima: Soudry and Carmon (2016)

Soudry and Carmon (2016):

- Mild over-parametrization $m_0 m_1 \geq n$, where m_l is the width of the activation layer $l \geq 1$; m_0 is the width of the input layer, $\boldsymbol{x} \in \mathbb{R}^{m_0}$.
- Gaussian dropout noise $\mathcal{E}$ and leaky-ReLU like activations
- Data $\boldsymbol{X} = [\boldsymbol{x}_1 \cdots \boldsymbol{x}_n], \boldsymbol{x}_i \in \mathbb{R}^{m_0}$, smoothed by small Gaussian noise
- Mild over-parametrization: $m_0 m_1 \geq n$, where m_l is the width of the activation layer l (for interpolation results, we assume $m_1 \geq n$)

Theorem (Single hidden layer) If $n \leq m_1 m_0$, then all differentiable local minima are global minima with MSE=0, $(\boldsymbol{X}, \mathcal{E})$ almost everywhere.

- They prove a similar result for the general depth (Theorem 5) assuming $m_{L-1} m_{L-2} \geq n$, where L is the total number of layers.
- Given the preceding results on global convergence, which assume extremely wide networks, there is considerable room for improving the convergence results to, say, $m_1 = O(n)$.

Over-parametrized NNs and Kernels

Cho&Saul (2009) and Tsuchida et al. (2018), and others:

- Exploit large width, m, to explicitly compute the kernel corresponding to a hidden layer

Notation

- $\boldsymbol{w}_i$ - independent random initial weights with density $f(w), w \in \mathbb{R}^n$
- $\boldsymbol{h}(\boldsymbol{x}), \boldsymbol{x} \in \mathbb{R}^n$ - hidden layer vector, m width of the hidden layer

$$\boldsymbol{h}(\boldsymbol{x}) := \frac{1}{\sqrt{m}}(\sigma(\langle \boldsymbol{w}_1, \boldsymbol{x}\rangle), \dots \sigma(\langle \boldsymbol{w}_m, \boldsymbol{x}\rangle))$$

Then, for $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ and large m

$$\begin{aligned} \langle \boldsymbol{h}(\boldsymbol{x}), \boldsymbol{h}(\boldsymbol{y})\rangle &= \frac{1}{m}\sum_{i=1}^{m} \sigma(\langle \boldsymbol{w}_i, \boldsymbol{x}\rangle)\sigma(\langle \boldsymbol{w}_i, \boldsymbol{y}\rangle) && (1) \\ &\approx \int_{\mathbb{R}^n} \sigma(\langle \boldsymbol{w}, \boldsymbol{x}\rangle)\sigma(\langle \boldsymbol{w}, \boldsymbol{y}\rangle) f(\boldsymbol{w}) d\boldsymbol{w} =: k(\boldsymbol{x}, \boldsymbol{y}) && (2) \end{aligned}$$

The "$\approx$" can be made precise in a probabilistic sense

Over-parametrization: Convergence to Global Minimum

Some notational conventions:

- Get rid of the bias, b: augment $\boldsymbol{x}$ with an auxiliary feature $x_0 \equiv 1$ and call $w_0 = b$, then
$$b + \langle \boldsymbol{w}, \boldsymbol{x} \rangle = \langle \boldsymbol{w}', \boldsymbol{x}' \rangle,$$
where $w' = (w_0, w_1, \ldots, w_d), x' = (x_0, x_1, \ldots, x_d)$. Hence, we get rid of b by embedding the problem in $d+1$ dimensions.
- For simplicity, data is often normalized on a hyper-sphere: $\|x\| = 1$
- For a 2-layer NN, weights in the second layer can be simplified:

$$f(\boldsymbol{W}, \boldsymbol{a}, \boldsymbol{x}) = \frac{1}{\sqrt{m}} \sum_{r=1}^{m} a_r \max(0, \langle \boldsymbol{w}_r, \boldsymbol{x} \rangle) = \frac{1}{\sqrt{m}} \sum_{r=1}^{m} \frac{a_r}{|a_r|} \max(0, \langle |a_r| w_r, \boldsymbol{x} \rangle)$$

  - Hence, $|a_r|$ can be incorporate into random weights
  - a_r can be assumed Bernoulli $\{\pm 1\}$ since $a_r/|a_r| \in \{\pm 1\}$
  - Scaling $1/\sqrt{m}$ is chosen for Law of Large Numbers as $(1/\sqrt{m})^2 = 1/m$
  - $\boldsymbol{W}$ is $d \times m$ matrix

Quadratic Loss and Gradient Descent (GD)

- **Training**: Given data set $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n, \boldsymbol{x}_i \in \mathbb{R}^d, y_i \in \mathbb{R}$. we minimize

$$\min_{\boldsymbol{W} \in \mathbb{R}^{d \times m}} L(\boldsymbol{W})$$

  for quadratic loss

$$L(\boldsymbol{W}) := \frac{1}{2} \sum_{i=1}^n (f(\boldsymbol{x}_i) - y_i)^2$$

- **Gradient:**

$$\frac{\partial L(\boldsymbol{W})}{\partial \boldsymbol{w}_r} = \frac{1}{\sqrt{m}} \sum_{i=1}^n (f(\boldsymbol{x}_i) - y_i) a_r \boldsymbol{x}_i 1_{\{\langle \boldsymbol{w}_r, \boldsymbol{x}_i \rangle \geq 0\}}$$

- **Gradient Descent (GD):** for step size $\eta > 0$

$$\boldsymbol{W}(k+1) = \boldsymbol{W}(k) - \eta \frac{\partial L(\boldsymbol{W}(k))}{\partial \boldsymbol{W}(k)}$$

Continuous Time Approximation

- **Gradient Flow:** Gradient descent with infinitesimal step size

$$\frac{\boldsymbol{W}(k+1) - \boldsymbol{W}(k)}{\eta} \approx \frac{d\boldsymbol{W}(t)}{dt} = -\frac{\partial L(\boldsymbol{W}(t))}{\partial \boldsymbol{W}(t)} \tag{3}$$

- Denote the prediction on input $\boldsymbol{x}_i$ at time t as

$$u_i(t) = f(W(t), \boldsymbol{a}, \boldsymbol{x}_i), \quad i = 1, \ldots, n$$

and

$$\boldsymbol{u}(t) = (u_1(t), \ldots, u_n(t))$$

Dynamical System

Time dynamics of each prediction

$$\begin{aligned}\frac{du_i(t)}{dt} &= \sum_{r=1}^{m} \left\langle \frac{\partial f(\boldsymbol{W}(t), \boldsymbol{x}_i)}{\partial \boldsymbol{w}_r(t)}, \frac{d\boldsymbol{w}_r(t)}{dt} \right\rangle \\ &= \sum_{j=1}^{n} (y_j - u_j) \sum_{r=1}^{m} \left\langle \frac{\partial f(\boldsymbol{W}(t), \boldsymbol{x}_i)}{\partial \boldsymbol{w}_r(t)}, \frac{\partial f(\boldsymbol{W}(t), \boldsymbol{x}_j)}{\partial \boldsymbol{w}_r(t)} \right\rangle \\ &=: \sum_{j=1}^{n} (y_j - u_j) H_{ij}(t)\end{aligned}$$

where in the second equality we used the gradient flow equation (3) and

$$H_{ij}(t) = \frac{1}{m} \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle \sum_{r=1}^{m} 1_{\{\langle \boldsymbol{x}_i, \boldsymbol{w}_r(t) \rangle \geq 0, \langle \boldsymbol{x}_j, \boldsymbol{w}_r(t) \geq 0 \rangle\}}$$

Define $n \times n$ matrix $\boldsymbol{H}(t) = \{H_{ij}(t)\}$

Dynamical System

The vector of predictors, $\boldsymbol{u}(t)$, evolves as

$$\frac{d}{dt}\boldsymbol{u}(t) = \boldsymbol{H}(t)(\boldsymbol{y} - \boldsymbol{u}(t)) \tag{4}$$

Objective: Prove that

$$\boldsymbol{u}(t) \to \boldsymbol{y}, \qquad \text{as } t \to \infty$$

for any

- initial random weights $\boldsymbol{w}_r = \boldsymbol{w}_r(0)$
- finite set of data points $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$

Key Ideas

- **Over-parametrization - large width** m, by the Laws of Large Numbers

$$H_{ij}(0) = \frac{1}{m}\langle \boldsymbol{x}_i, \boldsymbol{x}_j\rangle \sum_{r=1}^{m} 1_{\{\langle \boldsymbol{x}_i, \boldsymbol{w}_r(0)\rangle \geq 0, \langle \boldsymbol{x}_j, \boldsymbol{w}_r(0) \geq 0\rangle\}}$$
$$\approx \langle \boldsymbol{x}_i, \boldsymbol{x}_j\rangle \, \mathbb{E}\, 1_{\{\langle \boldsymbol{x}_i, \boldsymbol{w}_r(0)\rangle \geq 0, \langle \boldsymbol{x}_j, \boldsymbol{w}_r(0) \geq 0\rangle\}} =: H_{ij}^{\infty}$$

- Assume m so large, $m = \Omega(n^6)$, so that the initial random state does not change much during training, $\boldsymbol{W}(t) \approx \boldsymbol{W}$, and therefore

$$\boldsymbol{H}(t) \approx \boldsymbol{H}(0) \approx \boldsymbol{H}^{\infty}$$

- Hence, the vector of predictors, $\boldsymbol{u}(t)$, evolves as

$$\frac{d}{dt}\boldsymbol{u}(t) = \boldsymbol{H}(t)(\boldsymbol{y} - \boldsymbol{u}(t)) \approx \boldsymbol{H}^{\infty}(\boldsymbol{y} - \boldsymbol{u}(t))$$

linear system with constant (time invariant) coefficients

Key Ideas: Minimum Eigenvalue

For the convergence of the linear system

$$\frac{d}{dt}\boldsymbol{u}(t) \approx \boldsymbol{H}^{\infty}(\boldsymbol{y} - \boldsymbol{u}(t))$$

- One needs the minimum eigenvalue

  $$\lambda_0 := \lambda_{\min}(\boldsymbol{H}^{\infty}) > 0$$

  which is assumed in the paper. Recall Cho&Saul (2009) result

  $$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \mathbb{E}\, 1_{\{\langle \boldsymbol{x}_i, \boldsymbol{w}_r)\rangle \geq 0, \langle \boldsymbol{x}_j, \boldsymbol{w}_r \geq 0\rangle\}} = \frac{1}{2} - \frac{1}{2\pi}\cos^{-1}\left(\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle)\right)$$

  Hence, one should have $\lambda_0 := \lambda_{\min}(\boldsymbol{H}^{\infty}) > 0$ if there is no parallel pair $\boldsymbol{x}_i, \boldsymbol{x}_j$, which, for $\|\boldsymbol{x}_i\|_2 = 1$, reduces to all points being distinct, $\boldsymbol{x}_i \neq \boldsymbol{x}_j$.

Formal Theorem

Assumptions:

- Positive minimum eigenvalue: $\lambda_0 := \lambda_{\min}(\boldsymbol{H}^\infty) > 0$
- Scaled data: $\|\boldsymbol{x}_i\|_2 = 1$ and $|y_i| \leq C$
- Over-parametrization:
$$m = \Omega\left(\frac{n^6}{\lambda_0^4 \delta^3}\right)$$
- Random initialization: $\boldsymbol{w}_r \sim N(\boldsymbol{0}, \boldsymbol{I})$ and $a_r \sim$ Uniform$(\{\pm 1\})$

Theorem 3.2 Then, on a set of probability at least $1 - \delta$

$$\|\boldsymbol{u}(t) - \boldsymbol{y}\|_2^2 \leq \exp(-\lambda_0 t)\|\boldsymbol{u}(0) - \boldsymbol{y}\|_2^2$$

Over-parametrization and Generalization

Fine-Grained Analysis of Optimization and Generalization for Overparameterized Two-Layer Neural Networks, by Arora et al., 2019.

- Refine the analysis of Du et al. (2019), but assume even wider networks - see Theorem 4.1

$$m = \Omega\left(\frac{n^7}{\lambda_0^4 \kappa^2 \delta^4 \epsilon^2}\right).$$

- Prove a generalization bound, in Theorem 5.1, that doesn't depend on m. Is this surprising?

Neural Tangent Kernel

- Today, we'll focus on the following paper:
  Neural Tangent Kernel: Convergence and Generalization in Neural Networks, by A. Jacot, F. Gabriel and C. Hongler, NIPS 2018.
- This paper considered similar questions of convergence as Du et al. (2019).
- They introduced a terminology of Neural Tangent Kernel (NTK)
- Well follow the notation from the paper:
  - N data points: $(\boldsymbol{x}_i, y_i), \boldsymbol{x}_i \in \mathbb{R}^d, y_i \in \mathbb{R}, 1 \leq i \leq N$.
  - Find the best fit of a cost $C(\cdot)$ over a function class, say quadratic

$$\min_{f \in \mathcal{F}} C(f) = \min_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} (f(\boldsymbol{x}_i) - y_i)^2.$$

Lineal Model

- Choose m features $f^{(i)} \in \mathcal{F}$
- Let a parametrized function $F : \mathbb{R}^P \to \mathcal{F}$

$$F(\boldsymbol{a}) := f(\boldsymbol{a}) = \frac{1}{\sqrt{P}} \sum_{i=1}^{P} a_i f^{(i)}$$

- We can choose $f^{(i)}$ i.i.d., such that the kernel is given by $\mathbb{E}[f^{(i)}(x) f^{(i)}(x')] = K(x, x')$.
- Then, for a given cost function C, we optimize the composition $C \circ F$, using, say, gradient descent. We find the optimal linear parameters $\boldsymbol{a}$.
- Convex problem when C is convex.

Neural Networks: Nonlinear Model

- L layers, $l = 0$ input and $l = L$ output; each hidden layer having $n_l, 1 \leq l \leq L - 1$ neurons
- Activation function $\sigma : \mathbb{R} \to \mathbb{R}$, say ReLU, $\sigma(x) = x_+$
- $W^{(\ell)}$ are $n_\ell \times n_{\ell+1}$ matrices connecting the layers, $\ell = 0, \dots, L - 1$.
- Total parameters: $\theta = (W^{(0)}, \dots, W^{(L-1)})$.
- Non-linear parametrization, netwrok function $f_\theta(x)$

$$
\begin{aligned}
\alpha(0)(x;\theta) &= x \\
\tilde{\alpha}^{(\ell+1)}(x;\theta) &= \frac{1}{\sqrt{n_\ell}} W^{(\ell)} \alpha^{(\ell)}(x;\theta) + \beta b^{(\ell)} \quad (\beta \in \mathbb{R}+) \quad \text{(preactivations)} \\
\alpha^{(\ell)}(x;\theta) &= \sigma(\tilde{\alpha}^{(\ell)}(x;\theta)) \quad \text{(activations)} \\
f_\theta &= \tilde{\alpha}^{(L)}(x;\theta)
\end{aligned}
$$

- Now, the loss $C \circ F^{(L)}$ is not convex
- Linearize the model by studying the infinite-width limit $(n_1, \dots, n_{L-1}) \to \infty$

Gaussian Initialization

- Initialize the parameters $\theta \sim \mathcal{N}(\mathbf{0}, I_P)$, where P is the number of parameters $P = \sum_{\ell=0}^{L-1} (n_\ell + 1) n_{\ell+1}$

  Proposition 1 In the infinite width limit $(n_1, \ldots, n_{L-1}) \to \infty$ the preactivations $\tilde{\alpha}_i^{(\ell)}(\cdot; \theta) : \mathbb{R}^{n_\ell} \to \mathbb{R}$ are i.i.d. Gaussian processes with covariance $\Sigma^{(\ell)}$ given by

$$\Sigma^{(1)}(x, y) = \frac{1}{n_0} x \cdot y + \beta^2$$
$$\Sigma^{(\ell+1)}(x, y) = \mathop{\mathbb{E}}_{\alpha \sim \mathcal{N}(\mathbf{0}, \Sigma^{(\ell)})} [\sigma(\alpha(x)) \sigma(\alpha(y))] + \beta^2$$

  In particular, f_θ is a Gaussian process with covariance $\Sigma^{(L)}$
  The **proof** follows by induction, conditioning and then passing width to infinity, i.e., conditioning on $\tilde{\alpha}^{(L)}$

$$\begin{aligned}\tilde{\Sigma}^{(L+1)}(x, y) &= \frac{1}{n_L} \alpha^{(L)}(x; \theta) \cdot \alpha^{(L)}(y; \theta) + \beta^2 \\ &\to \mathop{\mathbb{E}}_{\alpha \sim \mathcal{N}(\mathbf{0}, \Sigma^{(L)})} [\sigma(\alpha(x)) \sigma(\alpha(y))] + \beta^2\end{aligned}$$

Training Neural Tangent Kernel (NTK)

- Gradient descent: path of steepest descent for quadratic loss

$$\partial_t \theta_p = -\partial_\theta (C \circ F^{(L)}) = \frac{2}{N} \sum_{i=1}^{N} (y_i - f_\theta(x_i)) \partial_{\theta_p} f_\theta(x_i)$$

  N - number of data points, for short $\partial_v = \partial / \partial v$.

- Evolution of $f_\theta(x)$:

$$\begin{aligned}\partial_t f_\theta(x) &= \sum_{p=1}^{P} \partial_{\theta_p} f_\theta(x) \partial_t \theta_p \quad \text{(chain rule)} \\ &= \frac{2}{N} \sum_{i=1}^{N} (y_i - f_\theta(x_i)) \left(\sum_{p=1}^{P} \partial_{\theta_p} f_\theta(x_i) \partial_{\theta_p} f_\theta(x) \right)\end{aligned}$$

- Neural Tangent Kernel (NTK):

$$\Theta^{(L)}(x, x') := \sum_{p=1}^{P} \partial_{\theta_p} f_\theta(x) \partial_{\theta_p} f_\theta(x')$$

  Note that this quantity was used implicitly in Du et al., and was denoted as $H(t)$

Infinite Width Limit of NTK

For finite width, $\Theta^{(L)}(t, x, x')$ is random, but it is deterministic in the infinite width limit. (This limit was H^∞ in Du et al.)

Theorem 1 For any $t < T$, as $(n_1, \ldots, n_{L-1}) \to \infty$,

$$\Theta^{(L)}(t) \to \Theta^{(L)}_\infty,$$

where

$$\Theta^{(L)}_\infty(x, x') = \sum_{\ell=1}^{L} \Sigma^{(\ell)}(x, x') \dot{\Sigma}^{(\ell+1)}(x, x') \cdots \dot{\Sigma}^{(L)}(x, x')$$

and

$$\dot{\Sigma}^{(\ell)}(x, x') = \mathop{\mathbb{E}}_{\alpha \sim \mathcal{N}(\mathbf{0}, \Sigma^{(\ell-1)})} [\dot{\sigma}(\alpha(x)) \dot{\sigma}(\alpha(x'))]$$

Why the Name NTK? - Tangent Model

Consider any parametric model $f(w, x), w \in \mathbb{R}^p, x \in \mathbb{R}^d$. Then, assuming f is differentiable, we can linearize this model around the initial parameters w_0 as:

$$f_0(w, x) := f(w_0, x) + (w - w_0) \cdot \nabla_w f(w_0, x), \quad \text{(Tangent model).} \quad (5)$$

Consider n data points $(x_i, y_i) \in \mathbb{R}^{d+1}, 1 \le i \le n$, and assume a quadratic loss, then training $f_0(w, x)$ is equivalent to solving a dual problem in the dot-product space according to tangent kernel (NTK):

$$K(x, y) = \nabla_w f(w_0, x) \cdot \nabla_w f(w_0, y).$$

Question: When is training $f_0(w, x)$ going to produce approximately the same results as $f(w, x)$?
Obviously, if $f(w, x)$ is linear, then $f_0(w, x) = f(w, x)$.

NTK: Main Result

Theorem 2 Assume non-polynomial activation σ, which is twice differentiable with bounded second derivative. Then, for least squares regression (and more), for any fixed T, the dynamics of gradient descent for $t \in [0, T]$ converges to the infinite width limit.

Key ingredient in the proof - Lemma 1 The weights $W^{(\ell)}(t)$ during training do not move much, or more precisely

$$\lim_{n_L \to \infty} \cdots \lim_{n_1 \to \infty} \sup_{t \in [0,T]} \left\| \frac{1}{\sqrt{n_\ell}} \left(W^{(\ell)}(t) - W^{(\ell)}(0) \right) \right\| = 0,$$

where $\| \cdot \|$ is the operator norm.

Reflection On Recent Convergence Papers

We have seen several papers that prove convergence, where

- Key ingredient is the weights do not move during training, i.e.,

  $$W(t) \approx W(0)$$

  This often referred to as Lazy Training.
- In essence, these models behave like linear models corresponding to the infinite with Gaussian kernel - Neural Tangent Kernel, or more precisely random sampling from this infinite width kernel.
- Are we missing something?
  - The main characteristics of NNs is that they are nonlinear models. If we are reducing them to linear Kernel models, why not work with kernels in the first place?
- What is the main property of the considered models that is constraining the weights from moving far?
  - Is it over-parametrization?
  - More on this next week...

Lazy Training

The following paper pinpoints the key ingredient responsible for wights remaining close to the initial values during training

- A Note on Lazy Training in Supervised Differentiable Programming, by L. Chizat and F. Bach, Dec 2018.

  See also an updated version:
  On Lazy Training Differentiable Programming, by L. Chizat and F. Bach, NIPS 2019.

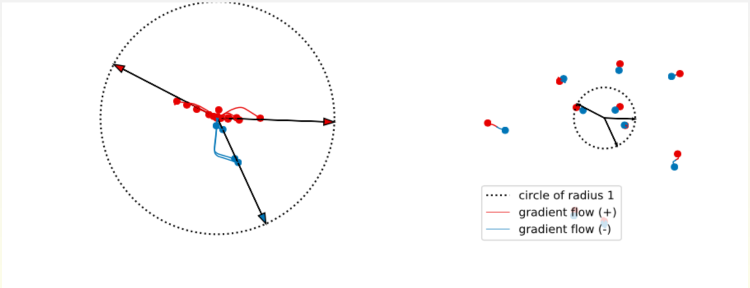
This paper puts in perspective many of the papers that we covered recently:

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In Advances in neural information processing systems, 2018.
Simon S. Du, Xiyu Zhai, Barnabs Pczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. In International Conference on Learning Representations, 2019.
Sanjeev Arora, Simon S. Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. arXiv preprint arXiv:1904.11955, 2019.
Simon S. Du, Lee Jason D., Li Haochuan, Wang Liwei, and Zhai Xiyu. Gradient descent finds global minima of deep neural networks. In International Conference on Machine Learning (ICML), 2019.
Yuanzhi Li and Yingyu Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. In Advances in Neural Information Processing Systems, pages 81678176, 2018.
Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In Proceedings of the 36th International Conference on Machine Learning, volume 97, pages 242252, 2019. Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu. Stochastic gradient descent optimizes over-parameterized deep ReLU networks. Machine Learning Journal, 2019.

The Answer Is In Scaling

They show that the key part of the model which ensures that the weights do not move far from the initial position $(w_{ij}(0))$ is the scaling $\alpha = 1/\sqrt{m}$, where m is the width of the network.

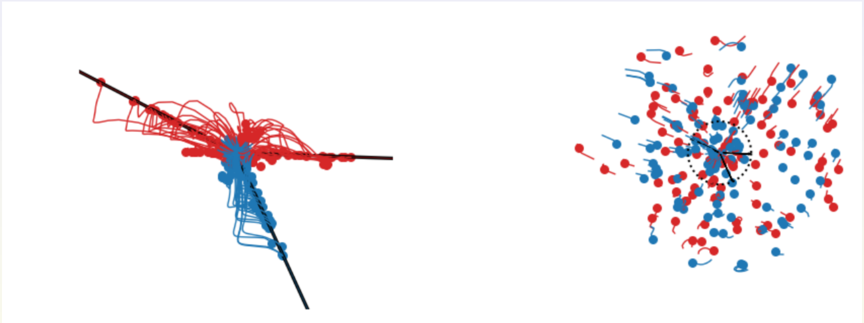
- Hence, the key to NTK approximation works because of scaling, rather than over-parametrization.
- They argue: any model can be scaled so that weights do not move much
  Example: τ - variance of $w_{ij}(0)$; "double trick" $f(x, w(0)) = 0$
  Ground truth: x uniform on a unit sphere, y output of NN with 3 neurons
  Model: $m = 20$ neurons - not a wide network



(a) "Active" training ($\tau = 0.1$) (b) Lazy training ($\tau = 2$)

Many Neurons Example

- Ground truth: x uniform on a unit sphere, y output of NN with 3 neurons
- $n = 200$ data samples
- Model wide network: $m = 200$ neurons
- τ - variance of $w_{ij}(0)$



(a) "Active" training ($\tau = 0.1$) (b) Lazy ($\tau = 2$)

Hence, the key to NTK regime is the scaling, rather than over-parametrization.

Recall Hoeffding's Bound

Lemma Let Z_i be i.i.d. random variables with $\mathbb{E}\, Z_i = \mu$ and bounded support $\mathbb{P}[a \leq Z_i \leq b] = 1$. Then, for any $\epsilon > 0$,

$$\mathbb{P}\left[\left|\frac{1}{m}\sum_{i=1}^{m} Z_i - \mu\right| > \epsilon\right] \leq 2\exp(-2m\epsilon^2/(b-a)^2)$$

Proof: Let's center Z_i: $X_i = Z_i - \mu$ and set $\bar{X} = (1/m)\sum X_i$. Then, for $\epsilon > 0$ and $\lambda > 0$, by Markov's inequality and i.i.d.

$$\mathbb{P}[\bar{X} \geq \epsilon] = \mathbb{P}[e^{\lambda\bar{X}} \geq e^{\lambda\epsilon}] \leq e^{-\lambda\epsilon}\,\mathbb{E}[e^{\lambda\bar{X}}] = e^{-\lambda\epsilon}\left(\mathbb{E}[e^{\lambda X_1/m}]\right)^m \tag{6}$$

Next, we show that

$$\mathbb{E}[e^{\lambda X_1}] \leq e^{\lambda^2(b-a)^2/8} \tag{7}$$

Let $a' = a - \mu$, $b' = b - \mu$ and note that $b' - a' = b - a$. By convexity

$$\mathbb{E}[e^{\lambda X_1}] \leq \frac{b' - \mathbb{E}[X_1]}{b-a}e^{\lambda a'} + \frac{\mathbb{E}[X] - a'}{b-a}e^{\lambda b'} = \frac{b'}{b-a}e^{\lambda a'} - \frac{a'}{b-a}e^{\lambda b'} =: f(\lambda)$$

Useful Tools: Hoeffding's Bound

Proof: Next, if $h = \lambda(b-a)$ and $p = -a'/(b-a)$, then, by Taylor's theorem

$$L(h) := \log(f(h/(b-a))) = -hp + \log(1 - p + pe^h) \leq \frac{h^2}{8}$$

since $L(0) = L'(0) = 0$and $L''(h) \leq 1/4$ for all h; this proves (7). Now, using (7) in (6), we obtain

$$\mathbb{P}[\bar{X} \geq \epsilon] \leq e^{-\lambda\epsilon}\,\mathbb{E}[e^{\lambda\bar{X}}] = e^{-\lambda\epsilon}\left(\mathbb{E}[e^{\lambda X_1/m}]\right)^m \leq e^{-\lambda\epsilon + \frac{\lambda^2(b-a)^2}{8m}},$$

which is minimized for

$$\lambda^* = \frac{4me}{(b-a)^2}$$

This concludes the proof of Hoeffding's bound.

Introduction to Generalization Error

The main objective of statistical learning is to predict well on future data.

- How do we measure the error?
  - Regression: Quadratic error/loss
$$\ell(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2$$
  - Classification: $\ell(\hat{y}, y) = 1_{\{\hat{y} \neq y\}}$
- $\{\mathcal{H}\}$: Hypothesis class of functions, e.g., all functions, $h(x, w)$, that can be generated by a NN of a certain architecture.
- Empirical Risk/Loss - total training error: For a data sample $S = \{(x_i, y_i)\}_{i=1}^{n}$ and $h \in \mathcal{H}$

$$\hat{L}_n = \hat{L}_n(h) \equiv L_S(h) = \frac{1}{n}\sum_{i=1}^{n} \ell(h(x_i), y_i)$$

Shai's book [ML2014] uses $L_S(h)$ notation; we'll use these interchangeably.

Empirical Risk Minimization and True Error

During training one typically minimizes the empirical risk (ERM), and obtain $\hat{h}_n \equiv h_S$

$$\hat{h}_n \in \arg\min_{h \in \mathcal{H}} \hat{L}(h)$$

True Risk=Population Risk

- Let $x \in \mathcal{X}, y \in \mathcal{Y}$, say $\mathcal{X} \subset \mathbb{R}^d, \mathcal{Y} \subset \mathbb{R}$, $z = (x, y) \in \mathcal{Z} = \mathcal{X} \times \mathcal{Y}$
- Define a probability measure on $\mathcal{Z}$, denoted by $\mathcal{D}$ in [ML2014] book
- Let $\{z_i = (x_i, y_i)\}_{i=1}^n$ be i.i.d. random variables on a product probability space $\mathcal{Z}^n$.
- Then, for $h \in \mathcal{H}$, we define a true risk/population risk as

$$L(h) \equiv L_{\mathcal{D}}(h) := \mathop{\mathbb{E}}_{x,y} \ell(h(x), y)$$

Probably Approximately Correct (PAC) Learning

The ultimate goal is to find h that minimizes the true risk, i.e.,

$$h \in \arg\min_{h \in \mathcal{H}} L(h)$$

But this is often impossible, leading to the more relaxed definition

> **Definition** (PAC Learnability) A hypothesis class $\mathcal{H}$ is PAC learnable with respect to a set $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ and a loss function $\ell : \mathcal{H} \times \mathcal{Z} \to \mathbb{R}_+$, if there exists a learning algorithm, which returns $\hat{h} \equiv \hat{h}(S)$ for a sample S of size $|S| = m$ with the following property: for every $0 < \epsilon, \delta < 1$ there exists $m(\epsilon, \delta)$, such that for all $m \geq m(\epsilon, \delta)$,
>
> $$\mathbb{P}\left[\left\{S \in \mathcal{Z}^m \;:\; L(\hat{h}(S)) \leq \min_{h \in \mathcal{H}} L(h) + \epsilon\right\}\right] \;\geq\; 1 - \delta$$

- PAC learning was introduced by Leslie Valiant (1984), who won for it the Turing Award in 2010.

Representative Sample

Definition (ϵ-representative sample)

A training set S is called ϵ-representative if

$$\forall h \in \mathcal{H}, \qquad |\hat{L}_S(h) - L(h)| \le \epsilon \ .$$

- Typically, we can only expect this to be true with high probability.

Representative Sample

Lemma

Assume that a training set S is $\frac{\epsilon}{2}$-representative. Then, any output of $\mathrm{ERM}_{\mathcal{H}}(S)$, namely any $\hat{h}_S \in \operatorname{argmin}_{h\in\mathcal{H}} \hat{L}_S(h)$, satisfies

$$L(\hat{h}_S) \;\leq\; \min_{h\in\mathcal{H}} L(h) + \epsilon \;.$$

Proof: For every $h \in \mathcal{H}$,

$$L(\hat{h}_S) \leq \hat{L}_S(\hat{h}_S) + \tfrac{\epsilon}{2} \leq \hat{L}_S(h) + \tfrac{\epsilon}{2} \leq L(h) + \tfrac{\epsilon}{2} + \tfrac{\epsilon}{2} = L(h) + \epsilon$$

- Hence, to ensure that ERM algorithm is a PAC (good) learner, we need to ensure that with probability $1-\delta$ a training set S is ϵ-representative

Uniform Convergence is Sufficient for Learnability

Definition (uniform convergence)

$\mathcal{H}$ has the *uniform convergence property* if there exists $m(\epsilon, \delta)$, such that for every $m \geq m(\epsilon, \delta)$, every distribution $\mathcal{D}$, and any i.i.d. sample $S \equiv S_m$ with distribution $\mathcal{D}$, S is ϵ-representatve with probability $1 - \delta$, i.e., if $m \geq m(\epsilon, \delta)$

$$\mathbb{P}\left[\left\{S \in \mathcal{Z}^m : \sup_{h \in \mathcal{H}} |\hat{L}_S(h) - L(h)| \leq \epsilon\right\}\right] \geq 1 - \delta$$

- A direct consequence of the preceding definition and lemma:

Corollary

- *If $\mathcal{H}$ has the uniform convergence property with a function $m(\epsilon, \delta)$ then $\mathcal{H}$ is PAC learnable with the sample complexity $m(\epsilon, \delta) \leq m(\epsilon/2, \delta)$.*
- *Furthermore, in that case, the $\mathrm{ERM}_{\mathcal{H}}$ paradigm is a successful agnostic PAC learner for $\mathcal{H}$.*

Finite Classes are PAC Learnable

We will prove the following:

Theorem

Assume $\mathcal{H}$ is finite and the range of the loss function is $[0,1]$. Then, $\mathcal{H}$ is agnostically PAC learnable using the $\mathrm{ERM}_{\mathcal{H}}$ algorithm with sample complexity

$$m_{\mathcal{H}}(\epsilon,\delta) \le \left\lceil \frac{2\log(2|\mathcal{H}|/\delta)}{\epsilon^2} \right\rceil .$$

Proof: It suffices to show that $\mathcal{H}$ has the uniform convergence property with

$$m_{\mathcal{H}}(\epsilon,\delta) \le \left\lceil \frac{\log(2|\mathcal{H}|/\delta)}{2\epsilon^2} \right\rceil .$$

Proof (cont.)

- To show uniform convergence, we need to show

$$\mathbb{P}(\{S : \sup_{h\in\mathcal{H}} |\hat{L}_S(h) - L(h)| > \epsilon\}) < \delta \ .$$

- Using the union bound:

$$\begin{aligned}\mathbb{P}(\max_{h\in\mathcal{H}} |\hat{L}_S(h) - L(h)| > \epsilon) &= \mathbb{P}(\cup_{h\in\mathcal{H}}\{|\hat{L}_S(h) - L(h)| > \epsilon\}) \\ &\le \sum_{h\in\mathcal{H}} \mathbb{P}(\{|\hat{L}_S(h) - L(h)| > \epsilon\}) \ .\end{aligned}$$

Proof (cont.)

- Recall: $z = (x, y), z_i = (x_i, y_i)$
- Recall: $L(h) = \mathbb{E}_z[\ell(h, z)]$ and $\hat{L}_S(h) = \frac{1}{m}\sum_{i=1}^m \ell(h, z_i)$.
- Denote $\theta_i = \ell(h, z_i)$.
- Then, for all i, $\mathbb{E}[\theta_i] = L(h)$

Then, by Hoeffding's inequality, since $0 \leq \theta_i \leq 1$, this implies:

$$\mathbb{P}[|\hat{L}_S(h) - L(h)| > \epsilon] \;\leq\; 2\,\exp\left(-2\,m\,\epsilon^2\right) \;\; .$$

Which, in conjunction with the union bound, results in

$$\mathbb{P}[\sup_{h \in \mathcal{H}} |\hat{L}_S(h) - L(h)| > \epsilon] \;\leq\; 2\,|\mathcal{H}|\,\exp\left(-2\,m\,\epsilon^2\right)$$

So, if $m \geq \frac{\log(2|\mathcal{H}|/\delta)}{2\epsilon^2}$ then the right-hand side is at most δ as required. $\square$

The Discretization Trick

- Suppose $\mathcal{H}$ is parameterized by d numbers
- Suppose we are happy with a representation of each number using b bits (say, $b = 32$)
- Then $|\mathcal{H}| \leq 2^{db}$, and so

$$m_{\mathcal{H}}(\epsilon, \delta) \leq \left\lceil \frac{2db + 2\log(2/\delta)}{\epsilon^2} \right\rceil .$$

- While not very elegant or tight, it's a great tool for upper bounding sample complexity

Reading

- PAC Learning and Generalization Theory: Chapters: 2-6 in [ML2014]
- Lazy vs Active Training
  - Neural Tangent Kernel: Convergence and Generalization in Neural Networks, by A. Jacot, F. Gabriel and C. Hongler, NIPS 2018.
  - A Note on Lazy Training in Supervised Differentiable Programming, by L. Chizat and F. Bach, Dec 2018.

    An updated version:
    On Lazy Training Differentiable Programming, by L. Chizat and F. Bach, NIPS 2019.
  - See also Chapter 8 in the recent monograph Deep Learning Theory Lecture Notes, by Telgarsky, Feb 2021.
- GD convergence
  - Gradient Descent Provably Optimizes Over-parameterized Neural Networks, by Du et al., ICLR, Feb 2019. Fine-Grained Analysis of Optimization and Generalization for Overparameterized Two-Layer Neural Networks, by Arora et al., Jan 2019.
- Local vs global minima
  - No bad local minima: Data independent training error guarantees for multilayer neural networks, by Soudry and Carmon, 2016.

Have Fun!