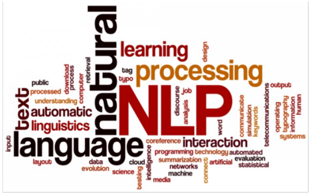
Towards Understanding Overparameterized Deep Neural Networks: From Optimization To Generalization

Quanquan Gu

Computer Science Department
UCLA

Joint work with Difan Zou, Yuan Cao and Dongruo Zhou

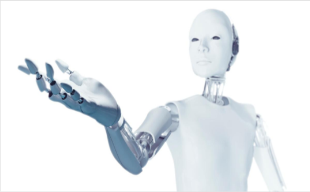
The Rise of Deep Learning



Natural Language Processing

Computer Vision

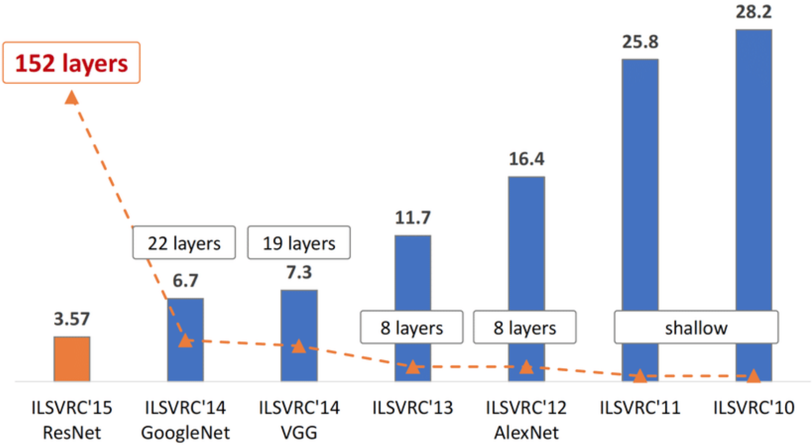
Go Games



Robots

The Rise of Deep Learning–Over-parameterization

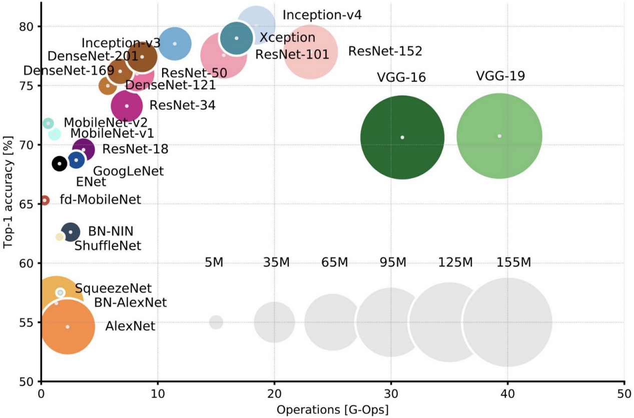
The evolution of the winning entries on the ImageNet



Alex Krizhevsky et al. 2012. "Imagenet classification with deep convolutional neural networks". In *Advances in neural information processing systems*, 1097–1105

The Rise of Deep Learning–Over-parameterization

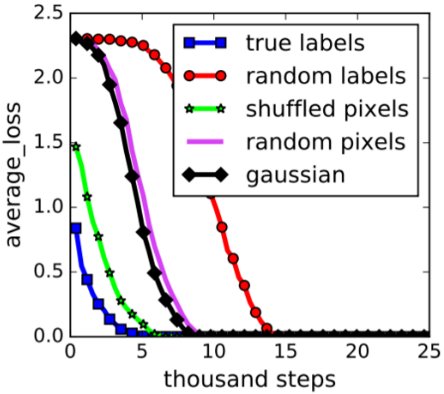
Top-1 accuracy versus amount of operations required for a single forward pass in popular neural networks.



Alfredo Canziani et al. 2016. "An analysis of deep neural network models for practical applications". *arXiv preprint arXiv:1605.07678*

Empirical Observations for DNNs

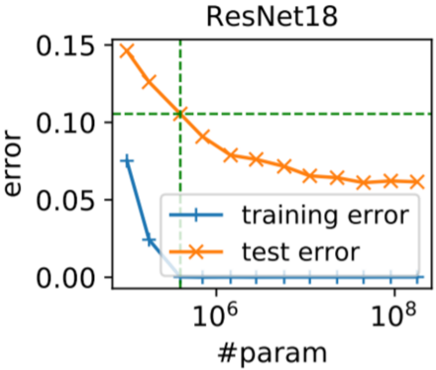
Fitting random labels and random pixels on CIFAR10



Chiyuan Zhang et al. 2016. "Understanding deep learning requires rethinking generalization". *arXiv preprint arXiv:1611.03530*

Empirical Observations for DNNs

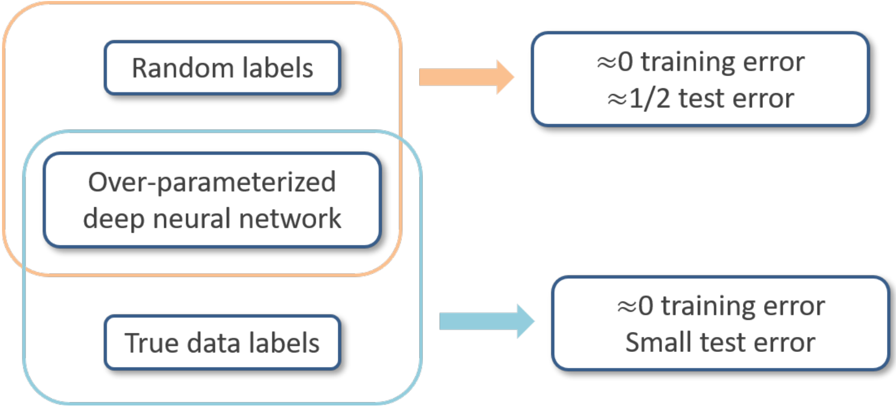
Training ResNet of different sizes on CIFAR10.



[1] Behnam Neyshabur et al. 2018. "Towards Understanding the Role of Over-Parametrization in Generalization of Neural Networks". *arXiv preprint arXiv:1805.12076*

Empirical Observations for DNNs

An empirical observation



Random labels

Over-parameterized deep neural network

True data labels

≈0 training error
≈1/2 test error

≈0 training error
Small test error

Optimization for DNNs

Question

Why and how does Over-parameterized DNNs trained by gradient descent can fit any labeling over distinct training inputs?

Challenge of Optimization for DNNs

Challenge

- The objective training loss is highly nonconvex or even nonsmooth.
- Conventional optimization theory can only guarantee finding second-order stationary point.

Existing Work

A line of research on the optimization theory for training two-layer NNs is limited to the teacher network setting (Goel et al. 2016; Tian 2017; Du et al. 2017; Li and Yuan 2017; Zhong et al. 2017; Zhang et al. 2018).

Limitations of these work:

- Assuming all data are generated from a "teacher network", which cannot be satisfied in practice.
- Strong assumptions on the training input (e.g., i.i.d. generated from Gaussian distribution).
- Requiring special initialization methods that are very different from the commonly-used one.

Existing Work

A line of research on the optimization theory for training two-layer NNs is limited to the teacher network setting (Goel et al. 2016; Tian 2017; Du et al. 2017; Li and Yuan 2017; Zhong et al. 2017; Zhang et al. 2018).

Limitations of these work:

- Assuming all data are generated from a “teacher network”, which cannot be satisfied in practice.
- Strong assumptions on the training input (e.g., i.i.d. generated from Gaussian distribution).
- Requiring special initialization methods that are very different from the commonly-used one.

Can we prove convergence under more practical assumptions?

Existing Work

Moreover, under much milder conditions on the training data and initialization, Li and Liang (2018) and Du et al. (2018b) established the global convergence of (stochastic) gradient descent for training two-layer ReLU networks.

- The neural network should be sufficiently over-parameterized (contains sufficiently large number of hidden nodes).
- The output of (stochastic) gradient descent can achieve abitrary small training loss.

Existing Work

Moreover, under much milder conditions on the training data and initialization, Li and Liang (2018) and Du et al. (2018b) established the global convergence of (stochastic) gradient descent for training two-layer ReLU networks.

- The neural network should be sufficiently over-parameterized (contains sufficiently large number of hidden nodes).
- The output of (stochastic) gradient descent can achieve abitrary small training loss.

Can similar results be generalized to DNNs?

Training Deep ReLU Networks

Setup

- Training data: $S = \{(\mathbf{x}_i, y_i)\}_{i=1,\dots,n}$ with input vector $\mathbf{x}_i \in \mathbb{R}^d$ and label $y_i \in \{-1,+1\}$. $\|\mathbf{x}_i\| = 1$, $(\mathbf{x}_i)_d = \mu$ and $\|\mathbf{x}_i - \mathbf{x}_j\|_2 \geq \phi$ if $y_i \neq y_j$.
- Fully connected ReLU network:

$$f_{\mathbf{W}}(\mathbf{x}) = \mathbf{v}^\top \sigma(\mathbf{W}_L^\top \sigma(\mathbf{W}_{L-1}^\top \cdots \sigma(\mathbf{W}_1^\top \mathbf{x}) \cdots))$$

  with weight matrices $\mathbf{W}_l \in \mathbb{R}^{m \times m}$ and $\mathbf{v} \in \{\pm 1\}^m$.
- Classifier: $\mathrm{sign}(y_i \cdot f_{\mathbf{W}}(\mathbf{x}_i))$
- Empirical risk minimization:

$$\min_{\mathbf{W}} L_S(\mathbf{W}) = \frac{1}{n}\sum_{i=1}^{n} \ell[y_i \cdot f_{\mathbf{W}}(\mathbf{x}_i)],$$

  where $\ell(z) = \log[1 + \exp(-z)]$.

Training Algorithm

Algorithm 1 (S)GD for training DNNs

```
input: Training data $\{\mathbf{x}_i, \mathbf{y}_i\}_{i\in[n]}$, step size $\eta$, total number of iterations $K$, minibatch size $B$.
initialization: $(\mathbf{W}_l^{(0)})_{i,j} \sim N(0, 2/m)$ for all $i, j, l$
—— Gradient Descent ——
for $k = 0, \ldots, K$ do
    $\mathbf{W}_l^{(k+1)} = \mathbf{W}_l^{(k)} - \eta \nabla_{\mathbf{W}_l} L_S(\mathbf{W}^{(k)})$ for all $l \in [L]$
end for
output: $\{\mathbf{W}_l^{(K)}\}_{l\in[L]}$
—— Stochastic Gradient Descent ——
for $k = 0, \ldots, K$ do
    Uniformly sample a minibatch of training data $\mathcal{B}^{(k)} \in [n]$
    $\mathbf{W}_l^{(k+1)} = \mathbf{W}_l^{(k)} - \frac{\eta}{B} \sum_{s\in\mathcal{B}^{(k)}} \nabla_{\mathbf{W}_l} \ell\big[y_i \cdot f_{\mathbf{W}^{(k)}}(\mathbf{x}_i)\big]$ for all $l \in [L]$
end for
output: $\{\mathbf{W}_l^{(K)}\}_{l\in[L]}$
```

Convergence of GD/SGD for Training DNNs

Theorem (Zou et al. 2018, informal)

For any $\epsilon > 0$, if

$$m = \widetilde{\Omega}\big(\textit{poly}(n, L, \phi^{-1}, \epsilon^{-1})\big)$$

then with high probability, (stochastic) gradient descent converges to a point that achieves ϵ-training loss within the following iteration number,

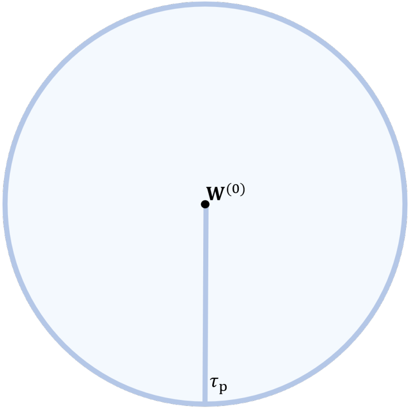
$$K = \mathcal{O}\big(\textit{poly}(n, L, \phi^{-1}, \epsilon^{-1})\big).$$

- The over-parameterization condition and iteration complexity are polynomial in all problem parameters.
- In order to achieve zero classification error, it suffices to set $\epsilon = \log(2)/n$.

Similar results have also been proved in Allen-Zhu et al. (2018b) and Du et al. (2018b) for regression problem with quadratic loss function.

Overview of Proof Technique

$$\mathcal{W}(\mathbf{W}^{(0)}, \tau) := \{\mathbf{W} = \{\mathbf{W}_l\}_{l=1}^{L} : \|\mathbf{W}_l - \mathbf{W}_l^{(0)}\|_F \leq \tau,\ l \in [L]\}$$

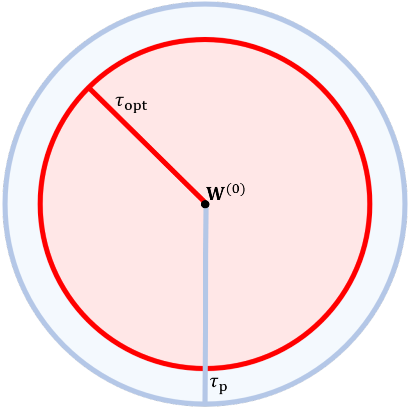


For large enough width m:

- For $\mathbf{W} \in \mathcal{W}(\mathbf{W}^{(0)}, \tau_{\mathrm{p}})$, $\tau_{\mathrm{p}} = \mathcal{O}(\text{poly}(n, L))$, $L_S(\mathbf{W})$ enjoys good curvature properties (e.g., sufficiently large gradient, nearly smooth and convex).

Overview of Proof Technique

$$\mathcal{W}(\mathbf{W}^{(0)}, \tau) := \{\mathbf{W} = \{\mathbf{W}_l\}_{l=1}^{L} : \|\mathbf{W}_l - \mathbf{W}_l^{(0)}\|_F \le \tau,\ l \in [L]\}$$



For large enough width m:

- For $\mathbf{W} \in \mathcal{W}(\mathbf{W}^{(0)}, \tau_{\mathrm{p}})$, $\tau_{\mathrm{p}} = \mathcal{O}(\mathrm{poly}(n, L))$, $L_S(\mathbf{W})$ enjoys good curvature properties (e.g., sufficiently large gradient, nearly smooth and convex).
- Gradient descent converges with trajectory length $\tau_{\mathrm{opt}} \le \mathcal{O}(\mathrm{poly}(n) \cdot \epsilon^{-1} m^{-1/2})$.
- Sufficiently large m can guarantee that $\tau_{\mathrm{opt}} \le \tau_p$.

Stronger Guarantees for Training DNNs

Theorem (Zou and Gu 2019, informal)

For any $\epsilon > 0$, if

$$m = \widetilde{\Omega}\big(n^8 L^{12} \phi^{-4}\big)$$

then with high probability, gradient descent converges to a point that achieves ϵ-training loss within the following iteration number,

$$K = \mathcal{O}\big(n^2 L^2 \phi^{-1} \log(1/\epsilon)\big).$$

Comparison with Existing Work

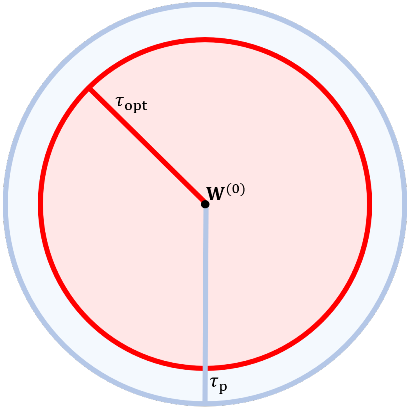
Table: Over-parameterization conditions and iteration complexities of GD for training deep neural networks.

	Over-para. condition	Iteration complexity	ReLU?
Du et al. (2018a)	$\Omega\left(\frac{2^{\mathcal{O}(L)} \cdot n^4}{\lambda_{\min}^4(\mathbf{K}^{(L)})}\right)$	$\mathcal{O}\left(\frac{2^{\mathcal{O}(L)} \cdot n^2 \log(1/\epsilon)}{\lambda_{\min}^2(\mathbf{K}^{(L)})}\right)$	no
Allen-Zhu et al. (2018b)	$\widetilde{\Omega}\left(\frac{n^{24} L^{12}}{\phi^8}\right)$	$\mathcal{O}\left(\frac{n^6 L^2 \log(1/\epsilon)}{\phi^2}\right)$	yes
Our work	$\widetilde{\Omega}\left(\frac{n^8 L^{12}}{\phi^4}\right)$	$\mathcal{O}\left(\frac{n^2 L^2 \log(1/\epsilon)}{\phi}\right)$	yes

$\mathbf{K}^{(L)}$ denotes the Gram matrix for L-hidden-layer neural network in Du et al. (2018a).

Overview of Proof Technique

$$\mathcal{W}(\mathbf{W}^{(0)}, \tau) := \{\mathbf{W} = \{\mathbf{W}_l\}_{l=1}^L : \|\mathbf{W}_l - \mathbf{W}_l^{(0)}\|_F \leq \tau,\ l \in [L]\}$$

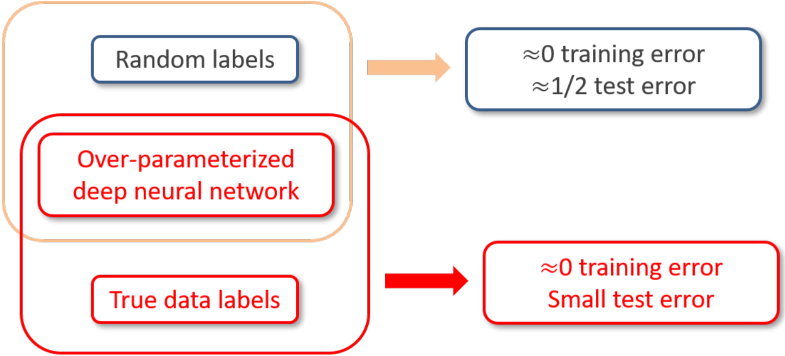


Improved techniques:

- Prove a larger τ_{p} such that for $\mathbf{W} \in \mathcal{W}(\mathbf{W}^{(0)}, \tau_{\text{p}})$, $L_S(\mathbf{W})$ enjoys good curvature properties.
- Within the region $\mathcal{W}(\mathbf{W}^{(0)}, \tau_{\text{p}})$, we prove larger gradient which leads to faster convergence of GD.
- We provide sharper characterization on the trajectory length of GD which leads to smaller τ_{opt}.
- Combine the above results we can significantly improve the condition on m to guarantee $\tau_{\text{opt}} \leq \tau_{\text{p}}$.

Empirical Observations for DNNs

An empirical observation



Optimization - Over-parameterized DNNs can fit ANY labeling over distinct training inputs.

Generalization - When the labeling is ‘nice’, over-parameterized DNNs can also be trained to achieve small test error.

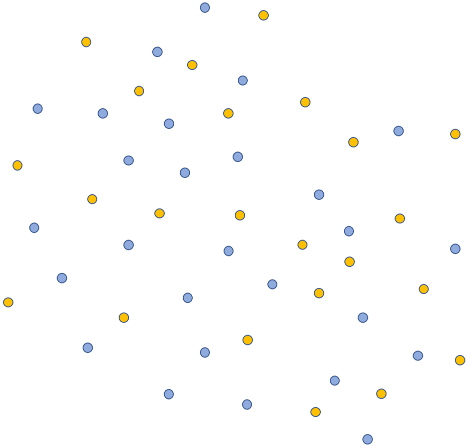
Limitations of Existing Generalization Bounds

Uniform convergence based generalization bounds (Neyshabur et al. 2015; Bartlett et al. 2017; Neyshabur et al. 2017; Golowich et al. 2017; Arora et al. 2018; Li et al. 2018; Wei et al. 2018) study

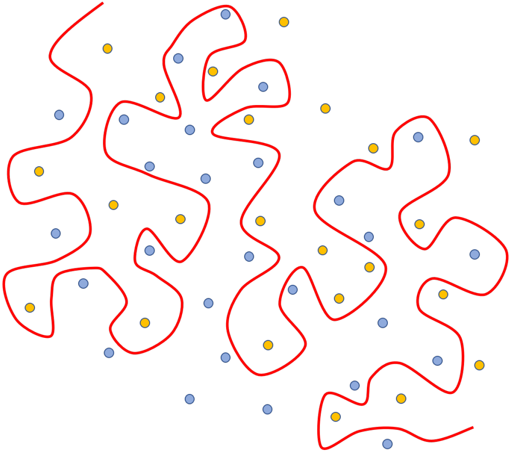
$$\sup_{f \in \mathcal{H}} \left| \underbrace{\frac{1}{n} \sum_{i=1}^{n} \ell[y_i \cdot f(\boldsymbol{x}_i)]}_{\text{training loss}} - \underbrace{\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \ell[y \cdot f(\mathbf{x})]}_{\text{test loss}} \right|$$

- Worst case analysis
- Trade-off between capacity (VC dimension, Rademacher complexity etc.) and training error

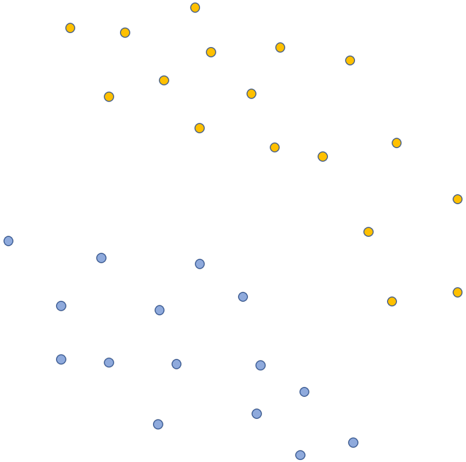
Wide Enough DNNs Can Fit Random Labels



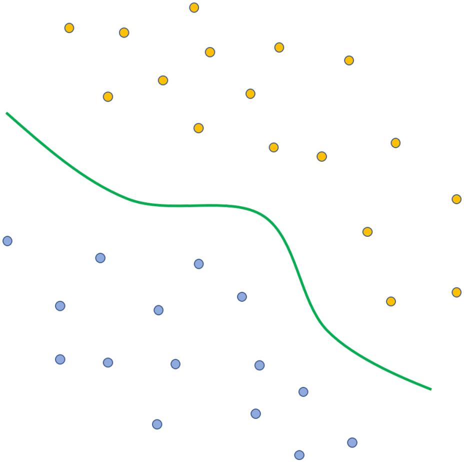
Wide Enough DNNs Can Fit Random Labels



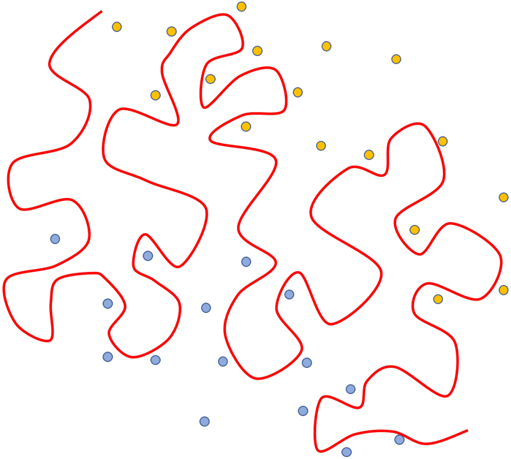
What If the Distribution is 'Nicer'?



What If the Distribution is 'Nicer'?

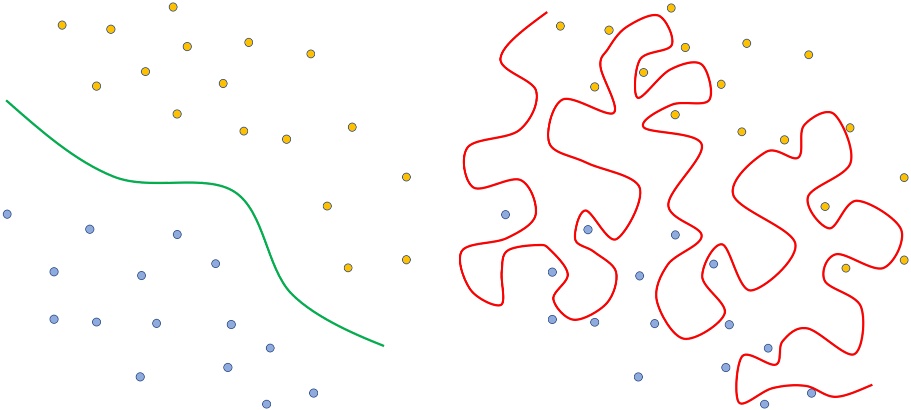


What If the Distribution is 'Nicer'?



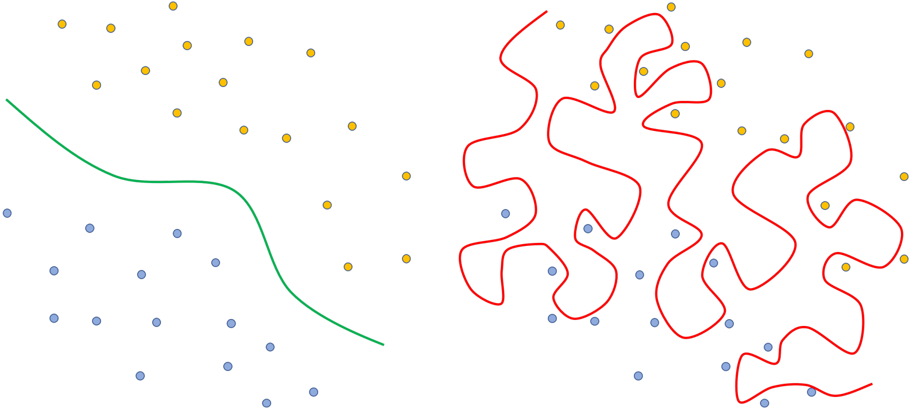
Limitations of Existing Generalization Bounds

Both neural networks separate the training data



Limitations of Existing Generalization Bounds

Both neural networks separate the training data



Algorithm-dependent generalization analysis is necessary

Algorithm-dependent Generalization Bounds

Algorithm-dependent bounds have been studied recently by (Li and Liang 2018; Allen-Zhu et al. 2018a; Arora et al. 2019a; Yehudai and Shamir 2019; E et al. 2019) for shallow networks.

Algorithm-dependent Generalization Bounds

Algorithm-dependent bounds have been studied recently by (Li and Liang 2018; Allen-Zhu et al. 2018a; Arora et al. 2019a; Yehudai and Shamir 2019; E et al. 2019) for shallow networks.

Questions haven't been fully answered:

- How to obtain generalization bounds for over-parameterized *deep* neural networks?
- How to quantify the 'classifiability' of the data distribution?
- What is the benefit of training each layer of the network?

Learning DNNs with Stochastic Gradient Descent

Setup

- Binary classification: $(\mathbf{x}, y) \in S^{d-1} \times \{\pm 1\}$ is generated from data distribution $\mathcal{D}$.
- Fully connected ReLU network:

$$f_{\mathbf{W}}(\mathbf{x}) = \sqrt{m} \cdot \mathbf{W}_L \sigma(\mathbf{W}_{L-1} \sigma(\mathbf{W}_{L-2} \cdots \sigma(\mathbf{W}_1 \mathbf{x}) \cdots)),$$

  where $\mathbf{W}_1 \in \mathbb{R}^{m \times d}$, $\mathbf{W}_l \in \mathbb{R}^{m \times m}$, $l = 2, \ldots, L-1$, $\mathbf{W}_L \in \mathbb{R}^{1 \times m}$, and $\sigma(\cdot) = \max\{\cdot, 0\}$.
- Expected risk minimization:

$$\min_{\mathbf{W}} L_{\mathcal{D}}(\mathbf{W}) := \mathbb{E}_{(\mathbf{x},y) \sim \mathcal{D}} \ell[y \cdot f_{\mathbf{W}}(\mathbf{x})],$$

  where $\ell(z) = \log[1 + \exp(-z)]$ is the cross-entropy loss.

Learning DNNs with Stochastic Gradient Descent

Algorithm 2 SGD for training DNNs

```
Input: Number of iterations $n$, step size $\eta$.
Generate each entry of $\mathbf{W}_l^{(0)}$ independently from $N(0, 2/m)$, $l \in [L-1]$.
Generate each entry of $\mathbf{W}_L^{(0)}$ independently from $N(0, 1/m)$.
for $i = 1, 2, \ldots, n$ do
    Draw $(\mathbf{x}_i, y_i)$ from $\mathcal{D}$.
    Update $\mathbf{W}^{(i)} = \mathbf{W}^{(i-1)} - \eta \cdot \nabla_{\mathbf{W}} \ell[y_i \cdot f_{\mathbf{W}^{(i-1)}}(\mathbf{x}_i)]$.
end for
Output: Randomly choose $\widehat{\mathbf{W}}$ uniformly from $\{\mathbf{W}^{(0)}, \ldots, \mathbf{W}^{(n-1)}\}$.
```

Neural Tangent Random Feature

Definition (Neural Tangent Random Feature)

Let $\mathbf{W}^{(0)}$ be generated via the initialization scheme in Algorithm 2. Define

$$\mathcal{F}(\mathbf{W}^{(0)}, R) = \big\{ f_{\mathbf{W}^{(0)}}(\cdot) + \langle \nabla_{\mathbf{W}} f_{\mathbf{W}^{(0)}}(\cdot), \mathbf{W} \rangle : \mathbf{W} \in \mathcal{W}(\mathbf{0}, R \cdot m^{-1/2}) \big\},$$

where $\mathcal{W}(\mathbf{W}, \tau) := \{\mathbf{W}' \in \mathcal{W} : \|\mathbf{W}_l' - \mathbf{W}_l\|_F \leq \tau, l \in [L]\}$.

Neural Tangent Random Feature

Definition (Neural Tangent Random Feature)

Let $\mathbf{W}^{(0)}$ be generated via the initialization scheme in Algorithm 2. Define

$$\mathcal{F}(\mathbf{W}^{(0)}, R) = \big\{ f_{\mathbf{W}^{(0)}}(\cdot) + \langle \nabla_{\mathbf{W}} f_{\mathbf{W}^{(0)}}(\cdot), \mathbf{W} \rangle : \mathbf{W} \in \mathcal{W}(\mathbf{0}, R \cdot m^{-1/2}) \big\},$$

where $\mathcal{W}(\mathbf{W}, \tau) := \{\mathbf{W}' \in \mathcal{W} : \|\mathbf{W}_l' - \mathbf{W}_l\|_F \leq \tau, l \in [L]\}$.

The training of the l-th layer of the network contributes the random features $\nabla_{\mathbf{W}_l} f_{\mathbf{W}^{(0)}}(\cdot)$ to the NTRF function class.

An Expected 0-1 Error Bound

Theorem (Cao and Gu 2019, informal)

For any $R > 0$*, if* $m \geq \widetilde{\Omega}\big(\mathrm{poly}(R, L, n)\big)$*, then with high probability, Algorithm 2 returns* $\widehat{\mathbf{W}}$ *that satisfies*

$$\mathbb{E}\big[L_{\mathcal{D}}^{0-1}(\widehat{\mathbf{W}})\big] \leq \inf_{f \in \mathcal{F}(\mathbf{W}^{(0)}, R)} \left\{ \frac{4}{n} \sum_{i=1}^{n} \ell[y_i \cdot f(\mathbf{x}_i)] \right\} + \mathcal{O}\left[\frac{LR}{\sqrt{n}} + \sqrt{\frac{\log(1/\delta)}{n}} \right].$$

An Expected 0-1 Error Bound

Theorem (Cao and Gu 2019, informal)

For any $R > 0$, if $m \geq \widetilde{\Omega}\big(\mathrm{poly}(R, L, n)\big)$, then with high probability, Algorithm 2 returns $\widehat{\mathbf{W}}$ that satisfies

$$\mathbb{E}\big[L_{\mathcal{D}}^{0-1}(\widehat{\mathbf{W}})\big] \leq \inf_{f \in \mathcal{F}(\mathbf{W}^{(0)}, R)} \left\{ \frac{4}{n} \sum_{i=1}^{n} \ell[y_i \cdot f(\mathbf{x}_i)] \right\} + \mathcal{O}\left[\frac{LR}{\sqrt{n}} + \sqrt{\frac{\log(1/\delta)}{n}} \right].$$

- Trade-off in the bound:
  - When R is small, first term is large, second term is small.
  - When R is large, first term is small, second term is large.

An Expected 0-1 Error Bound

Theorem (Cao and Gu 2019, informal)

For any $R > 0$, *if* $m \geq \widetilde{\Omega}\big(\mathrm{poly}(R, L, n)\big)$, *then with high probability, Algorithm 2 returns* $\widehat{\mathbf{W}}$ *that satisfies*

$$\mathbb{E}\big[L_{\mathcal{D}}^{0-1}(\widehat{\mathbf{W}})\big] \leq \inf_{f \in \mathcal{F}(\mathbf{W}^{(0)}, R)} \left\{ \frac{4}{n} \sum_{i=1}^{n} \ell[y_i \cdot f(\mathbf{x}_i)] \right\} + \mathcal{O}\left[\frac{LR}{\sqrt{n}} + \sqrt{\frac{\log(1/\delta)}{n}} \right].$$

- Trade-off in the bound:
  - When R is small, first term is large, second term is small.
  - When R is large, first term is small, second term is large.
- When $R = \widetilde{\mathcal{O}}(1)$, the second term is standard large-deviation error.

An Expected 0-1 Error Bound

Theorem (Cao and Gu 2019, informal)

For any $R > 0$, *if* $m \geq \widetilde{\Omega}\big(\mathrm{poly}(R, L, n)\big)$, *then with high probability, Algorithm 2 returns* $\widehat{\mathbf{W}}$ *that satisfies*

$$\mathbb{E}\big[L_{\mathcal{D}}^{0-1}(\widehat{\mathbf{W}})\big] \leq \inf_{f \in \mathcal{F}(\mathbf{W}^{(0)}, R)} \left\{ \frac{4}{n} \sum_{i=1}^{n} \ell[y_i \cdot f(\mathbf{x}_i)] \right\} + \mathcal{O}\left[\frac{LR}{\sqrt{n}} + \sqrt{\frac{\log(1/\delta)}{n}} \right].$$

- Trade-off in the bound:
  - When R is small, first term is large, second term is small.
  - When R is large, first term is small, second term is large.
- When $R = \widetilde{\mathcal{O}}(1)$, the second term is standard large-deviation error.
- Deep neural networks compete with the best function in the NTRF function class $\mathcal{F}(\mathbf{W}^{(0)}, \widetilde{\mathcal{O}}(1))$.

An Expected 0-1 Error Bound

Theorem (Cao and Gu 2019, informal)

For any $R > 0$*, if* $m \geq \widetilde{\mathcal{O}}\big(\mathrm{poly}(R, L, n)\big)$*, then with high probability, Algorithm 2 returns* $\widehat{\mathbf{W}}$ *that satisfies*

$$\mathbb{E}\big[L_{\mathcal{D}}^{0-1}(\widehat{\mathbf{W}})\big] \leq \inf_{f \in \mathcal{F}(\mathbf{W}^{(0)}, R)} \left\{ \frac{4}{n} \sum_{i=1}^{n} \ell[y_i \cdot f(\mathbf{x}_i)] \right\} + \mathcal{O}\left[\frac{LR}{\sqrt{n}} + \sqrt{\frac{\log(1/\delta)}{n}} \right].$$

The training of the l-th layer of the network enlarges $\mathcal{F}(\mathbf{W}^{(0)}, \widetilde{\mathcal{O}}(1))$, leading to smaller expected 0-1 loss.

An Expected 0-1 Error Bound

Theorem (Cao and Gu 2019, informal)

For any $R > 0$, *if* $m \geq \widetilde{\mathcal{O}}(\mathrm{poly}(R, L, n))$, *then with high probability, Algorithm 2 returns* $\widehat{\mathbf{W}}$ *that satisfies*

$$\mathbb{E}\big[L_{\mathcal{D}}^{0-1}(\widehat{\mathbf{W}})\big] \leq \inf_{f \in \mathcal{F}(\mathbf{W}^{(0)}, R)} \left\{ \frac{4}{n} \sum_{i=1}^{n} \ell[y_i \cdot f(\mathbf{x}_i)] \right\} + \mathcal{O}\left[\frac{LR}{\sqrt{n}} + \sqrt{\frac{\log(1/\delta)}{n}} \right].$$

The training of the l-th layer of the network enlarges $\mathcal{F}(\mathbf{W}^{(0)}, \widetilde{\mathcal{O}}(1))$, leading to smaller expected 0-1 loss.

The "classifiability" of the underlying data distribution $\mathcal{D}$ can be measured by how well its i.i.d. examples can be classified by $\mathcal{F}(\mathbf{W}^{(0)}, \widetilde{\mathcal{O}}(1))$.

Connection to Neural Tangent Kernel (NTK)

Definition (Neural Tangent Kernel)

The neural tangent kernel is defined as:

$$\boldsymbol{\Theta}^{(L)} = (\boldsymbol{\Theta}_{i,j}^{(L)})_{n\times n},\ \boldsymbol{\Theta}_{i,j}^{(L)} := m^{-1}\mathbb{E}\big[\langle \nabla_{\mathbf{W}} f_{\mathbf{W}^{(0)}}(\mathbf{x}_i), \nabla_{\mathbf{W}} f_{\mathbf{W}^{(0)}}(\mathbf{x}_j)\rangle\big].$$

- Consistent with definitions given in Jacot et al. (2018), Yang (2019), and Arora et al. (2019b).
- Fixes exponential dependence in L in the original definition in Jacot et al. (2018) by using $N(0, 2/m)$ initialization instead of $N(0, 1/m)$.

Connection to Neural Tangent Kernel (NTK)

Corollary (Cao and Gu 2019, informal)

Let $\mathbf{y} = (y_1, \dots, y_n)^\top$ *and* $\lambda_0 = \lambda_{\min}(\mathbf{\Theta}^{(L)})$*. If* $m \geq \widetilde{\Omega}\big(\mathrm{poly}(L, n, \lambda_0^{-1})\big)$*, then with high probability, Algorithm 2 returns* $\widehat{\mathbf{W}}$ *that satisfies*

$$\mathbb{E}\big[L_{\mathcal{D}}^{0-1}(\widehat{\mathbf{W}})\big] \leq \widetilde{\mathcal{O}}\left[L \cdot \sqrt{\frac{\mathbf{y}^\top (\mathbf{\Theta}^{(L)})^{-1} \mathbf{y}}{n}}\right] + \mathcal{O}\left[\sqrt{\frac{\log(1/\delta)}{n}}\right].$$

Connection to Neural Tangent Kernel (NTK)

Corollary (Cao and Gu 2019, informal)

Let $\mathbf{y} = (y_1, \ldots, y_n)^\top$ *and* $\lambda_0 = \lambda_{\min}(\mathbf{\Theta}^{(L)})$. *If* $m \geq \widetilde{\Omega}\big(\mathrm{poly}(L, n, \lambda_0^{-1})\big)$, *then with high probability, Algorithm 2 returns* $\widehat{\mathbf{W}}$ *that satisfies*

$$\mathbb{E}\big[L_{\mathcal{D}}^{0-1}(\widehat{\mathbf{W}})\big] \leq \widetilde{\mathcal{O}}\left[L \cdot \sqrt{\frac{\mathbf{y}^\top (\mathbf{\Theta}^{(L)})^{-1}\mathbf{y}}{n}}\right] + \mathcal{O}\left[\sqrt{\frac{\log(1/\delta)}{n}}\right].$$

The "classifiability" of the underlying data distribution $\mathcal{D}$ can also be measured by the quantity $\mathbf{y}^\top (\mathbf{\Theta}^{(L)})^{-1}\mathbf{y}$.

Overview of Proof Technique

Key observations

- Deep ReLU networks are *almost linear* in terms of their parameters in a small neighbourhood around random initialization

$$f_{\mathbf{W}'}(\mathbf{x}_i) \approx f_{\mathbf{W}}(\mathbf{x}_i) + \langle \nabla f_{\mathbf{W}}(\mathbf{x}_i), \mathbf{W}' - \mathbf{W} \rangle.$$

- $L_{(\mathbf{x}_i, y_i)}(\mathbf{W})$ is *Lipschitz continuous* and *almost convex*

$$\|\nabla_{\mathbf{W}_l} L_{(\mathbf{x}_i, y_i)}(\mathbf{W})\|_F \le \mathcal{O}(\sqrt{m}),\ l \in [L],$$

$$L_{(\mathbf{x}_i, y_i)}(\mathbf{W}') \gtrsim L_{(\mathbf{x}_i, y_i)}(\mathbf{W}) + \langle \nabla_{\mathbf{W}} L_{(\mathbf{x}_i, y_i)}(\mathbf{W}), \mathbf{W}' - \mathbf{W} \rangle.$$

Overview of Proof Technique

Key observations

- Deep ReLU networks are *almost linear* in terms of their parameters in a small neighbourhood around random initialization

$$f_{\mathbf{W}'}(\mathbf{x}_i) \approx f_{\mathbf{W}}(\mathbf{x}_i) + \langle \nabla f_{\mathbf{W}}(\mathbf{x}_i), \mathbf{W}' - \mathbf{W} \rangle.$$

- $L_{(\mathbf{x}_i, y_i)}(\mathbf{W})$ is *Lipschitz continuous* and *almost convex*

$$\|\nabla_{\mathbf{W}_l} L_{(\mathbf{x}_i,y_i)}(\mathbf{W})\|_F \leq \mathcal{O}(\sqrt{m}),\ l \in [L],$$
$$L_{(\mathbf{x}_i,y_i)}(\mathbf{W}') \gtrsim L_{(\mathbf{x}_i,y_i)}(\mathbf{W}) + \langle \nabla_{\mathbf{W}} L_{(\mathbf{x}_i,y_i)}(\mathbf{W}), \mathbf{W}' - \mathbf{W} \rangle.$$

Optimization for Lipschitz and convex functions

+

Online-to-batch conversion

Conclusion

Under certain data distribution assumptions

- Global convergence guarantees for GD in training over-parameterized deep ReLU networks
- SGD trains an over-parameterized deep ReLU network and achieves $\widetilde{\mathcal{O}}(n^{-1/2})$ expected 0-1 loss.
- The data "classifiability" can be measured by the NTRF function class or the NTK kernel matrix.
- An algorithm-dependent generalization error bound.
- Sample complexity is independent of network width.

Future Work

- Deeper understanding of the NTRF function class and NTK.
- Other learning algorithms.
- Other neural network architectures.

Thank you!

References I

Allen-Zhu, Zeyuan, Yuanzhi Li, and Yingyu Liang. 2018a. "Learning and Generalization in Overparameterized Neural Networks, Going Beyond Two Layers". *arXiv preprint arXiv:1811.04918*.

Allen-Zhu, Zeyuan, Yuanzhi Li, and Zhao Song. 2018b. "A Convergence Theory for Deep Learning via Over-Parameterization". *arXiv preprint arXiv:1811.03962*.

Arora, Sanjeev, et al. 2019a. "Fine-Grained Analysis of Optimization and Generalization for Overparameterized Two-Layer Neural Networks". *arXiv preprint arXiv:1901.08584*.

Arora, Sanjeev, et al. 2019b. "On exact computation with an infinitely wide neural net". *arXiv preprint arXiv:1904.11955*.

Arora, Sanjeev, et al. 2018. "Stronger generalization bounds for deep nets via a compression approach". *arXiv preprint arXiv:1802.05296*.

Bartlett, Peter L, Dylan J Foster, and Matus J Telgarsky. 2017. "Spectrally-normalized margin bounds for neural networks". In *Advances in Neural Information Processing Systems*, 6240–6249.

Canziani, Alfredo, Adam Paszke, and Eugenio Culurciello. 2016. "An analysis of deep neural network models for practical applications". *arXiv preprint arXiv:1605.07678*.

Cao, Yuan, and Quanquan Gu. 2019. "Generalization Bounds of Stochastic Gradient Descent for Wide and Deep Neural Networks". *arXiv preprint arXiv:1905.13210*.

References II

Du, Simon S, Jason D Lee, and Yuandong Tian. 2017. “When is a Convolutional Filter Easy To Learn?” *arXiv preprint arXiv:1709.06129*.

Du, Simon S, et al. 2018a. “Gradient Descent Finds Global Minima of Deep Neural Networks”. *arXiv preprint arXiv:1811.03804*

Du, Simon S, et al. 2018b. “Gradient Descent Provably Optimizes Over-parameterized Neural Networks”. *arXiv preprint arXiv:1810.02054*.

E, Weinan, Chao Ma, and Lei Wu. 2019. “A comparative analysis of the optimization and generalization property of two-layer neural network and random feature models under gradient descent dynamics”. *arXiv preprint arXiv:1904.04326*.

Goel, Surbhi, et al. 2016. “Reliably learning the relu in polynomial time”. *arXiv preprint arXiv:1611.10258*.

Golowich, Noah, Alexander Rakhlin, and Ohad Shamir. 2017. “Size-Independent Sample Complexity of Neural Networks”. *arXiv preprint arXiv:1712.06541*.

Jacot, Arthur, Franck Gabriel, and Clément Hongler. 2018. “Neural Tangent Kernel: Convergence and Generalization in Neural Networks”. *arXiv preprint arXiv:1806.07572*.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton. 2012. “Imagenet classification with deep convolutional neural networks”. In *Advances in neural information processing systems*, 1097–1105.

Li, Xingguo, et al. 2018. “On Tighter Generalization Bound for Deep Neural Networks: CNNs, ResNets, and Beyond”. *arXiv preprint arXiv:1806.05159*.

References III

Li, Yuanzhi, and Yingyu Liang. 2018. "Learning overparameterized neural networks via stochastic gradient descent on structured data". *arXiv preprint arXiv:1808.01204*.

Li, Yuanzhi, and Yang Yuan. 2017. "Convergence Analysis of Two-layer Neural Networks with ReLU Activation". *arXiv preprint arXiv:1705.09886*.

Neyshabur, Behnam, Ryota Tomioka, and Nathan Srebro. 2015. "Norm-based capacity control in neural networks". In *Conference on Learning Theory*, 1376–1401.

Neyshabur, Behnam, et al. 2017. "A pac-bayesian approach to spectrally-normalized margin bounds for neural networks". *arXiv preprint arXiv:1707.09564*.

Neyshabur, Behnam, et al. 2018. "Towards Understanding the Role of Over-Parametrization in Generalization of Neural Networks". *arXiv preprint arXiv:1805.12076*.

Tian, Yuandong. 2017. "An Analytical Formula of Population Gradient for two-layered ReLU network and its Applications in Convergence and Critical Point Analysis". *arXiv preprint arXiv:1703.00560*.

Wei, Colin, et al. 2018. "On the margin theory of feedforward neural networks". *arXiv preprint arXiv:1810.05369*.

Yang, Greg. 2019. "Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation". *arXiv preprint arXiv:1902.04760*.

References IV

Yehudai, Gilad, and Ohad Shamir. 2019. “On the power and limitations of random features for understanding neural networks”. *arXiv preprint arXiv:1904.00687*.

Zhang, Chiyuan, et al. 2016. “Understanding deep learning requires rethinking generalization”. *arXiv preprint arXiv:1611.0353*

Zhang, Xiao, et al. 2018. “Learning One-hidden-layer ReLU Networks via Gradient Descent”. *arXiv preprint arXiv:1806.07808*.

Zhong, Kai, et al. 2017. “Recovery Guarantees for One-hidden-layer Neural Networks”. *arXiv preprint arXiv:1706.03175*.

Zou, Difan, and Quanquan Gu. 2019. “An Improved Analysis of Training Over-parameterized Deep Neural Networks”. *arXiv preprint arXiv:1906.04688*.

Zou, Difan, et al. 2018. “Stochastic gradient descent optimizes over-parameterized deep ReLU networks”. *arXiv preprint arXiv:1811.08888*.