Towards Understanding Overparameterized Deep Neural Networks: From Optimization To Generalization

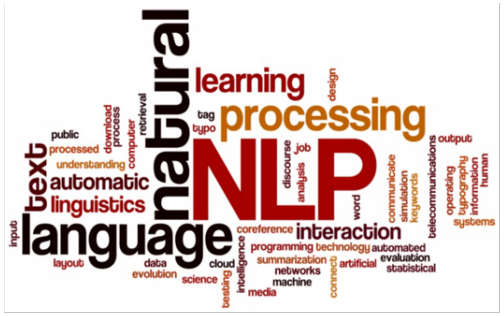
Quanquan Gu

Computer Science Department
UCLA

Joint work with Difan Zou, Yuan Cao and Dongruo Zhou

TTIC Workshop on Recent Trends in Clustering and **Classification**

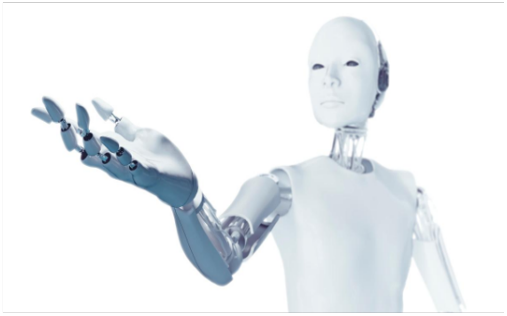
The Rise of Deep Learning



Natural Language Processing

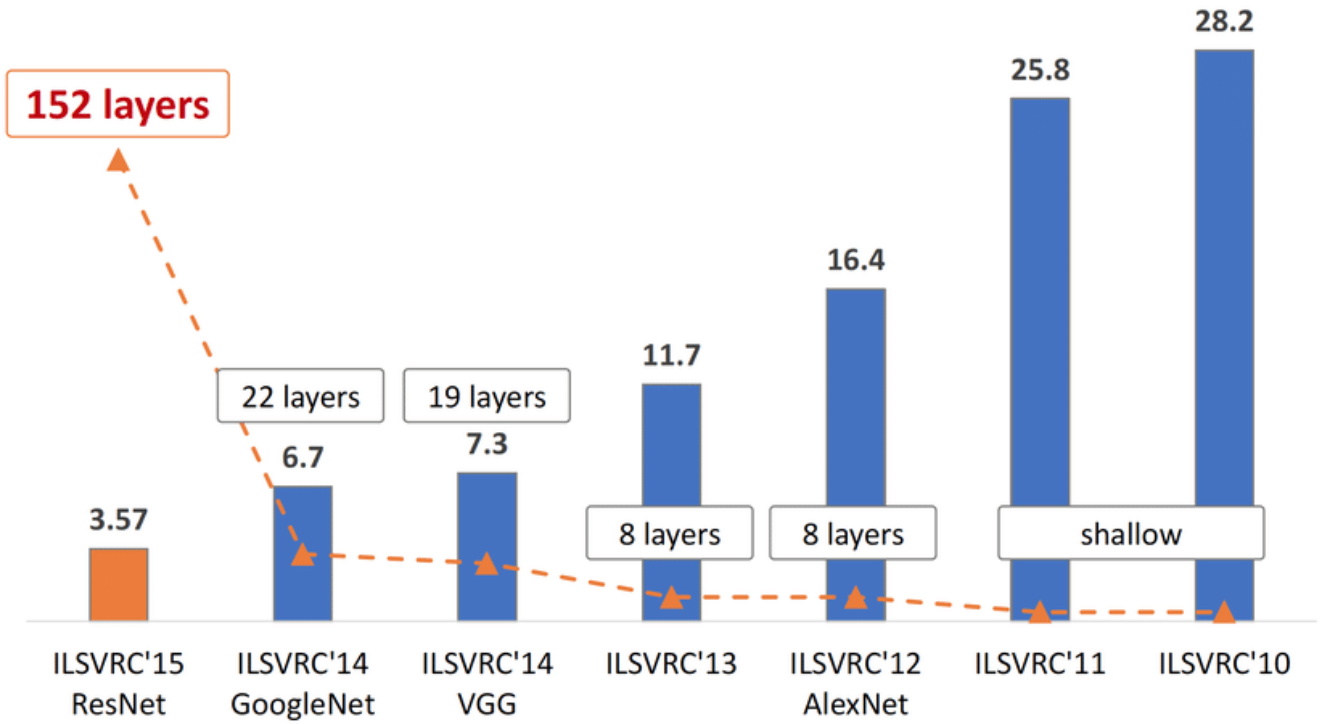
Computer Vision

Go Games



Robots

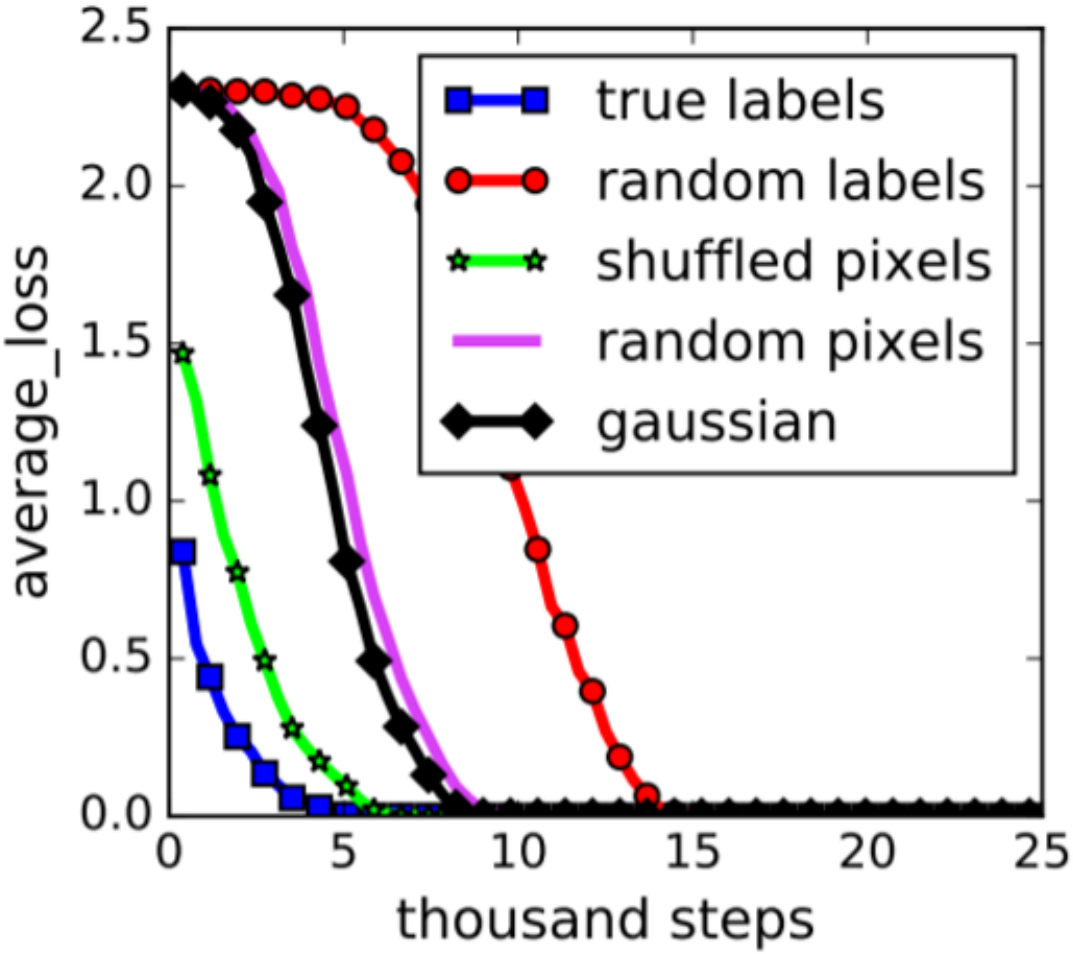
The evolution of the winning entries on the ImageNet



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

Empirical Observations for DNNs

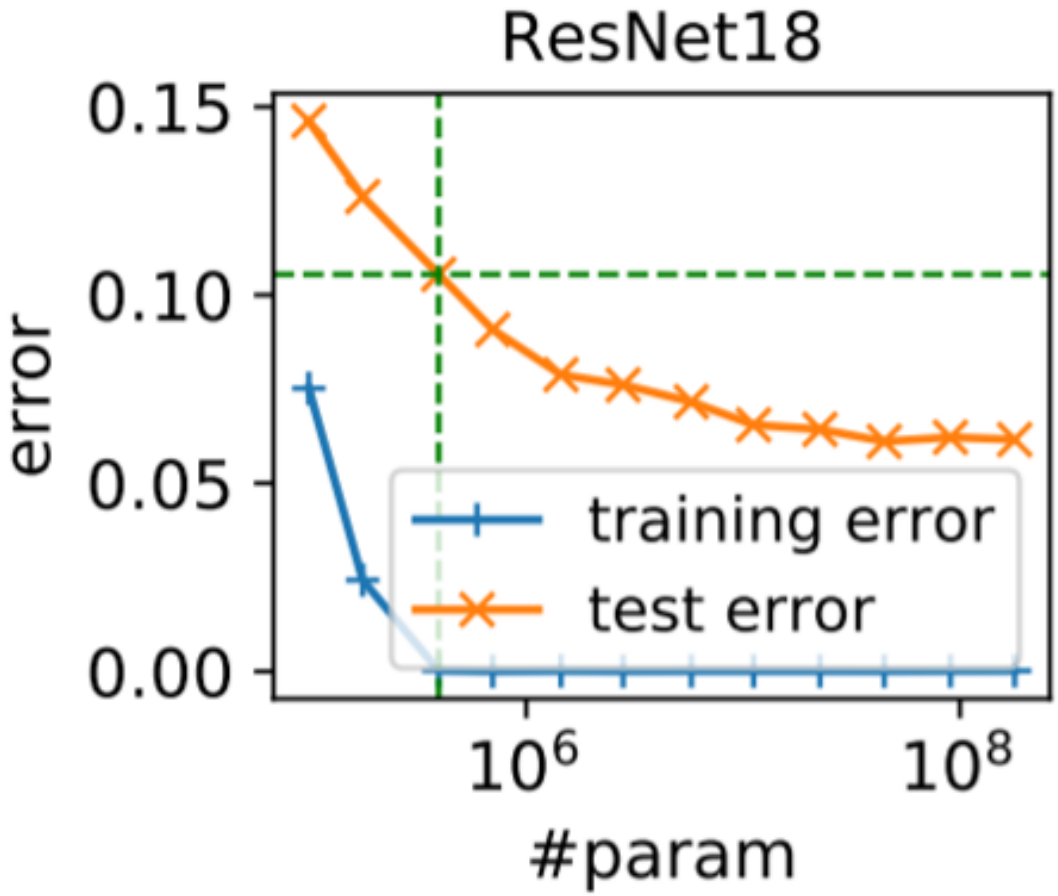
Fitting random labels and random pixels on CIFAR10



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.



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

Optimization of Neural Networks

- Fully connected neural 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 matrix $\mathbf{W}_l \in \mathbb{R}^{m \times m}$ and $\mathbf{v} \in \{\pm 1\}^m$.
- $\sigma(\cdot)$ is the activation function, e.g., ReLU: $\sigma(t) = \max(0, t)$, sigmoid, etc.
- Given a training sample $S = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$,

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

- $\ell(\cdot, \cdot)$ denotes the loss function, e.g., cross-entropy loss, square loss, etc.

Optimization of Neural Networks

Question 1

Why over-parameterized neural networks trained by gradient descent can fit training data with arbitrary labels?

Optimization of Neural Networks

Question 1

Why over-parameterized neural networks trained by gradient descent can fit training data with arbitrary labels?

Challenges:

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

Optimization of Shallow Networks

Early Work:

- A line of research (Goel et al. 2016; Tian 2017; Du et al. 2017; Li and Yuan 2017; Zhong et al. 2017; Zhang et al. 2018) on two-layer neural networks assumes a underlying teacher network, and essentially does parameter recovery

Optimization of Shallow Networks

Early Work:

- A line of research (Goel et al. 2016; Tian 2017; Du et al. 2017; Li and Yuan 2017; Zhong et al. 2017; Zhang et al. 2018) on two-layer neural networks assumes a underlying teacher network, and essentially does parameter recovery

Recent Work:

- **Li and Liang (2018):** When the data comes from mixtures of well-separated distributions, stochastic gradient descent (SGD) learns two-layer ReLU networks for multi-class classification problem (using cross-entropy loss) if the neural network width satisfies $m = \Omega(\mathrm{poly}(l, k, \epsilon^{-1}))$ (l: number of mixtures, k: number of classes, ϵ target expected error).
- **Du et al. (2018):** When the training data matrix is not degenerate (i.e., no duplicate data), gradient descent (GD) optimizes two-layer ReLU networks for regression problem (using square loss) if the neural network width satisfies $m = \Omega(n^6/\lambda_0^4)$ (n: sample size, λ_0: smallest eigenvalue of some gram matrix).

Optimization of Shallow Networks

Early Work:

- A line of research (Goel et al. 2016; Tian 2017; Du et al. 2017; Li and Yuan 2017; Zhong et al. 2017; Zhang et al. 2018) on two-layer neural networks assumes a underlying teacher network, and essentially does parameter recovery

Recent Work:

- **Li and Liang (2018):** When the data comes from mixtures of well-separated distributions, stochastic gradient descent (SGD) learns two-layer ReLU networks for multi-class classification problem (using cross-entropy loss) if the neural network width satisfies $m = \Omega(\mathrm{poly}(l, k, \epsilon^{-1}))$ (l: number of mixtures, k: number of classes, ϵ target expected error).
- **Du et al. (2018):** When the training data matrix is not degenerate (i.e., no duplicate data), gradient descent (GD) optimizes two-layer ReLU networks for regression problem (using square loss) if the neural network width satisfies $m = \Omega(n^6/\lambda_0^4)$ (n: sample size, λ_0: smallest eigenvalue of some gram matrix).

How to prove the convergence
of GD/SGD for training DNNs?

Optimization of Deep Networks

- **Allen-Zhu et al. (2018):** When any two data points are well separated, both GD and SGD optimize deep ReLU networks for regression problem (using square loss) if the neural network width satisfies $m = \widetilde{\Omega}(\text{poly}(L, n))$ (n: sample size, L: neural network depth).
- **Du et al. (2018):** When the training data matrix is not degenerate, GD optimizes deep neural networks with smooth activation functions for regression problem (using square loss) if the neural network width satisfies $m = \Omega\big(\text{poly}(n)2^{O(L)}\big)$.
- **Zou et al. (2018):** When any two data from different classes are well separated, both GD and SGD optimize deep ReLU networks for binary classification problem (with a class of loss functions) if the neural network width satisfies $m = \widetilde{\Omega}(\text{poly}(L, n))$.

Convergence of GD for Training DNNs

Theorem (Zou et al. (2018), Informal)

If any two data points from different classes are separated by a constant ϕ and the neural network width satisfies

$$m = \widetilde{\Omega}\big(n^{14} L^{16} / \phi^4\big)$$

then with high probability, gradient descent converges to a point that achieves zero training error within the following iteration number,

$$K = O\big(n^5 L^3 / \phi\big).$$

- The over-parameterization condition and iteration complexity are polynomial in all problem parameters.

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

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. (2018)	$\Omega\left(\frac{2^{O(L)}\cdot n^4}{\lambda_{\min}^4(\mathbf{K}^{(L)})}\right)$	$O\left(\frac{2^{O(L)}\cdot n^2\log(1/\epsilon)}{\lambda_{\min}^2(\mathbf{K}^{(L)})}\right)$	no
Allen-Zhu et al. (2018)	$\widetilde{\Omega}\left(\frac{n^{24}L^{12}}{\phi^8}\right)$	$O\left(\frac{n^6L^2\log(1/\epsilon)}{\phi^2}\right)$	yes
Zou et al. (2018)	$\widetilde{\Omega}\left(\frac{n^{14}L^{16}}{\phi^4}\right)$	$O\left(\frac{n^5L^3}{\phi}\right)$	yes

$\mathbf{K}^{(L)}$ is the conjugate kernel (Daniely (2017)) for L-hidden-layer neural network.

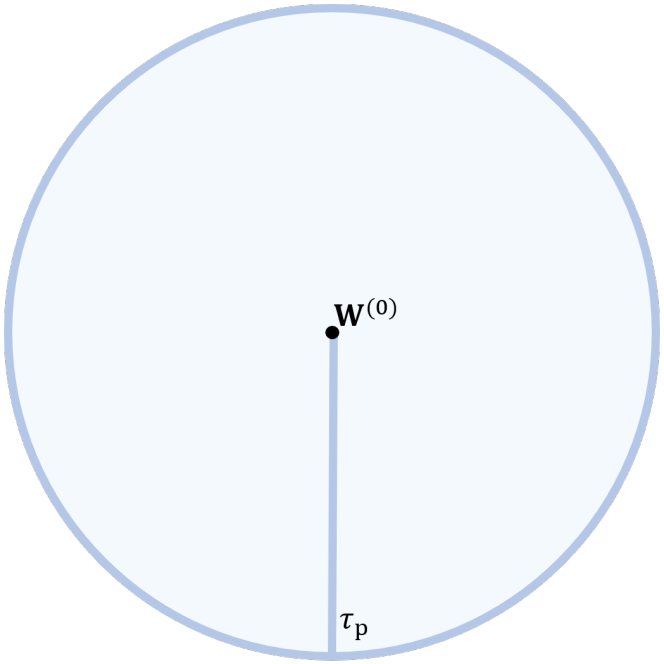
Comparison with Existing Work

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

	Over-para. condition	Iteration complexity	ReLU?
Du et al. (2018)	$\Omega\left(\frac{2^{O(L)}\cdot n^4}{\lambda_{\min}^4(\mathbf{K}^{(L)})}\right)$	$O\left(\frac{2^{O(L)}\cdot n^2\log(1/\epsilon)}{\lambda_{\min}^2(\mathbf{K}^{(L)})}\right)$	no
Allen-Zhu et al. (2018)	$\widetilde{\Omega}\left(\frac{n^{24}L^{12}}{\phi^8}\right)$	$O\left(\frac{n^6L^2\log(1/\epsilon)}{\phi^2}\right)$	yes
Zou et al. (2018)	$\widetilde{\Omega}\left(\frac{n^{14}L^{16}}{\phi^4}\right)$	$O\left(\frac{n^5L^3}{\phi}\right)$	yes
Zou and Gu (2019)	$\widetilde{\Omega}\left(\frac{n^8L^{12}}{\phi^4}\right)$	$O\left(\frac{n^2L^2\log(1/\epsilon)}{\phi}\right)$	yes

Overview of Proof Technique

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

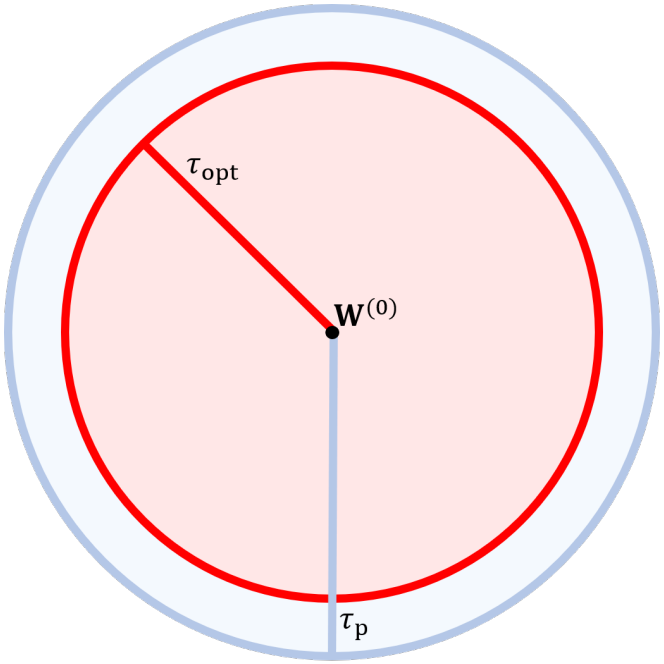


For large enough width m:

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

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

Generalization of Deep Neural Networks

Question 2

Why an over-parameterized neural network can generalize even when it interpolates the training data? What kind of data can be learned by over-parameterized networks?

Existing Work

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) study

- **Bartlett et al. (2017)** gives a Rademacher complexity based generalization error bound:

$$\widetilde{O}\bigg(\sqrt{\frac{m}{n}}\prod_{l=1}^{L}\|\mathbf{W}_l\|_2\bigg[\sum_{l=1}^{L}\frac{\|\mathbf{W}_l^\top-\mathbf{W}_l^{(0)\top}\|_{2,1}^{2/3}}{\|\mathbf{W}_l\|_2^{2/3}}\bigg]^{3/2}\bigg)$$

- **Neyshabur et al. (2018)** provides a bound using the PAC-Bayes framework

$$\widetilde{O}\bigg(L\sqrt{\frac{m}{n}}\prod_{l=1}^{L}\|\mathbf{W}_l\|_2\bigg[\sum_{l=1}^{L}\frac{(\sqrt{m}\|\mathbf{W}_l-\mathbf{W}_l^{(0)}\|_F)^2}{\|\mathbf{W}_l\|_2^2}\bigg]^{1/2}\bigg)$$

Existing Work

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) study

- **Bartlett et al. (2017)** gives a Rademacher complexity based generalization error bound:

$$\widetilde{O}\bigg(\sqrt{\frac{m}{n}}\prod_{l=1}^{L}\|\mathbf{W}_l\|_2\bigg[\sum_{l=1}^{L}\frac{\|\mathbf{W}_l^\top-\mathbf{W}_l^{(0)\top}\|_{2,1}^{2/3}}{\|\mathbf{W}_l\|_2^{2/3}}\bigg]^{3/2}\bigg)$$

- **Neyshabur et al. (2018)** provides a bound using the PAC-Bayes framework

$$\widetilde{O}\bigg(L\sqrt{\frac{m}{n}}\prod_{l=1}^{L}\|\mathbf{W}_l\|_2\bigg[\sum_{l=1}^{L}\frac{(\sqrt{m}\|\mathbf{W}_l-\mathbf{W}_l^{(0)}\|_F)^2}{\|\mathbf{W}_l\|_2^2}\bigg]^{1/2}\bigg)$$

Cannot provide generalization bounds independent of network width in the practical setting.

Algorithm-dependent Bounds for Shallow Networks

- **Li and Liang (2018):** When the data comes from mixtures of well-separated distributions, SGD learns two-layer ReLU networks for multi-class classification problem.
- **Allen-Zhu et al. (2018):** Over-parameterized three-layer ReLU networks can learn three-layer narrower networks with smooth activation functions when trained with SGD.
- **Arora et al. (2019):** GD can train a two-layer ReLU network with fixed second-layer weights to achieve a generalization error of the form $\widetilde{O}(\sqrt{\mathbf{y}^\top (\mathbf{H}^\infty)^{-1}\mathbf{y}}/n)$

Algorithm-dependent Bounds for Shallow Networks

- **Li and Liang (2018):** When the data comes from mixtures of well-separated distributions, SGD learns two-layer ReLU networks for multi-class classification problem.
- **Allen-Zhu et al. (2018):** Over-parameterized three-layer ReLU networks can learn three-layer narrower networks with smooth activation functions when trained with SGD.
- **Arora et al. (2019):** GD can train a two-layer ReLU network with fixed second-layer weights to achieve a generalization error of the form $\widetilde{O}(\sqrt{\mathbf{y}^\top (\mathbf{H}^\infty)^{-1}\mathbf{y}}/n)$

How to prove algorithm-dependent generalization bounds for DNNs?

Algorithm-dependent Bounds for Deep Networks

- **Daniely (2017):** A neural network of large enough size is competitive with the best function in the conjugate kernel class of the network. The training is essentially only on the last layer (the hidden layer updates are negligible).
- **Cao and Gu (2019):** Over-parameterized ReLU networks can compete with the best function in the *neural tangent random feature* function class. The generalization bound can also be written in the form $\widetilde{O}(\sqrt{\mathbf{y}^\top (\mathbf{\Theta}^{(L)})^{-1}\mathbf{y}}/n)$ ($\mathbf{y}$: label vector, $\mathbf{\Theta}^{(L)}$: gram matrix of neural tangent kernel (Jacot et al. 2018)).

Algorithm-dependent Bounds for Deep Networks

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, SGD returns $\widehat{\mathbf{W}}$ that satisfies

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

where

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

Algorithm-dependent Bounds for Deep Networks

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, SGD returns $\widehat{\mathbf{W}}$ that satisfies

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

where

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

- 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.

Algorithm-dependent Bounds for Deep Networks

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, SGD returns $\widehat{\mathbf{W}}$ that satisfies

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

where

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

- 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 $R = O(1)$, the second term is standard large-deviation error.

Algorithm-dependent Bounds for Deep Networks

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, SGD returns* $\widehat{\mathbf{W}}$ *that satisfies*

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

where $\mathbf{\Theta}^{(L)}$ is the neural tangent kernel (Jacot et al. 2018).

$\mathbf{\Theta}_{i,j}^{(L)} := \lim_{m\to\infty} m^{-1}\langle \nabla_{\mathbf{W}} f_{\mathbf{W}^{(0)}}(\mathbf{x}_i), \nabla_{\mathbf{W}} f_{\mathbf{W}^{(0)}}(\mathbf{x}_j)\rangle$.

Algorithm-dependent Bounds for Deep Networks

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, SGD returns* $\widehat{\mathbf{W}}$ *that satisfies*

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

where $\mathbf{\Theta}^{(L)}$ is the neural tangent kernel (Jacot et al. 2018).

$\mathbf{\Theta}_{i,j}^{(L)} := \lim_{m\to\infty} m^{-1}\langle \nabla_{\mathbf{W}} f_{\mathbf{W}^{(0)}}(\mathbf{x}_i), \nabla_{\mathbf{W}} f_{\mathbf{W}^{(0)}}(\mathbf{x}_j)\rangle$.

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*

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

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*

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

Optimization for Lipschitz and (almost) convex functions

+

Online-to-batch conversion

Summary

For wide enough deep neural networks

- GD and SGD can find global minima of training loss on regular training data with arbitrary labeling.
- Neural networks trained by SGD can achieve $\widetilde{O}\big(n^{-1/2}\big)$ generalization error if the training data admits small $\mathbf{y}^\top \big(\mathbf{\Theta}^{(L)}\big)^{-1}\mathbf{y}$.

Future Work and Open Problems

- For optimization, the best condition on the neural network width is $\widetilde{\Omega}(n^8)$ (Zou and Gu 2019), can we weaken this condition?
- For generalization, some recent results show that neural networks can beat kernel regression on handcrafted learning problems (Allen-Zhu et al. 2019, Wei et al. 2019). How to show it in the general setting?

Thank you!

References I

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

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.

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. 2018. "Gradient Descent Provably Optimizes Over-parameterized Neural Networks". *arXiv preprint arXiv:1810.02054*.

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*.

References II

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*.

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*.

References III

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

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*.