Towards a Foundation of Deep Learning: SGD, Overparametrization, and Generalization

Jason D. Lee

University of Southern California

January 29, 2019
Successes of Deep Learning

- Game-playing (AlphaGo, DOTA, King of Glory)
- Computer Vision (Classification, Detection, Reasoning.)
- Automatic Speech Recognition
- Natural Language Processing (Machine Translation, Chatbots)
Successes of Deep Learning

- Game-playing (AlphaGo, DOTA, King of Glory)
- Computer Vision (Classification, Detection, Reasoning.)
- Automatic Speech Recognition
- Natural Language Processing (Machine Translation, Chatbots)
Goal: A few steps towards theoretical understanding of Optimization and Generalization in Deep Learning.
1. Challenges

2. Saddlepoints and SGD

3. Landscape Design via Overparametrization

4. Algorithmic/Implicit Regularization
Theoretical Challenges: Two Major Hurdles

1. Optimization
   - Non-convex and non-smooth with exponentially many critical points.

2. Statistical
   - Successful Deep Networks are huge with more parameters than samples (overparametrization).
Theoretical Challenges: Two Major Hurdles

1. Optimization
   - Non-convex and non-smooth with exponentially many critical points.

2. Statistical
   - Successful Deep Networks are huge with more parameters than samples (overparametrization).

Two Challenges are Intertwined

Learning = Optimization Error + Statistical Error.
But Optimization and Statistics Cannot Be Decoupled.

- The choice of optimization algorithm affects the statistical performance (generalization error).
- Improving statistical performance (e.g. using regularizers, dropout …) changes the algorithm dynamics and landscape.
Practical observation: Gradient methods find high quality solutions.
Practical observation: Gradient methods find high quality solutions.

Theoretical Side: Even finding a local minimum is NP-hard!
Practical observation: Gradient methods find high quality solutions.

Theoretical Side: Even finding a local minimum is NP-hard!

Follow the Gradient Principle: No known convergence results for even back-propagation to stationary points!
Practical observation: Gradient methods find high quality solutions.

Theoretical Side: Even finding a local minimum is NP-hard!

Follow the Gradient Principle: No known convergence results for even back-propagation to stationary points!

Question

1. Why is (stochastic) gradient descent (GD) successful? Or is it just “alchemy”?
(Sub)-Gradient Descent

Gradient Descent algorithm:

\[ x_{k+1} = x_k - \alpha_k \partial f(x_k). \]

Non-smoothness

Deep Learning Loss Functions are not smooth! (e.g. ReLU, max-pooling, batch-norm)
Theorem (Davis, Drusvyatskiy, Kakade, and Lee)

Let $x_k$ be the iterates of the stochastic sub-gradient method. Assume that $f$ is locally Lipschitz, then every limit point $x^*$ is critical:

$$0 \in \partial f(x^*).$$

- Previously, convergence of sub-gradient method to stationary points is only known for weakly-convex functions $(f(x) + \frac{\lambda}{2} \|x\|^2$ convex). $(1 - \text{ReLU}(x))^2$ is not weakly convex.

- Convergence rate is polynomial in $\frac{\sqrt{d}}{\epsilon^4}$, to $\epsilon$-subgradient for a smoothing SGD variant.
Can subgradients be efficiently computed?

Automatic Differentiation a.k.a Backpropagation

Automatic Differentiation uses the chain rule with dynamic programming to compute gradients in time $5x$ of function evaluation.

**However, there is no chain rule for subgradients!**

$$x = \sigma(x) - \sigma(-x),$$

TensorFlow/Pytorch will give the wrong answer.
Can subgradients be efficiently computed?

Automatic Differentiation a.k.a Backpropagation

Automatic Differentiation uses the chain rule with dynamic programming to compute gradients in time $5x$ of function evaluation.

However, there is no chain rule for subgradients!

$$x = \sigma(x) - \sigma(-x),$$

TensorFlow/Pytorch will give the wrong answer.

Theorem (Kakade and Lee 2018)

There is a chain rule for subgradients. Using this chain rule with randomization, Automatic Differentiation can compute a subgradient in time $6x$ of function evaluation.
Theorem (Lee et al., COLT 2016)

Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be a twice continuously differentiable function with the strict saddle property, then gradient descent with a random initialization converges to a local minimizer or negative infinity.

- Theorem applies for many optimization algorithms including coordinate descent, mirror descent, manifold gradient descent, and ADMM (Lee et al. 2017 and Hong et al. 2018)
- Stochastic optimization with injected isotropic noise finds local minimizers in polynomial time (Pemantle 1992; Ge et al. 2015, Jin et al. 2017)
Why are local minimizers interesting?

All local minimizers are global and SGD/GD find the global min:

1. Overparametrized Networks with Quadratic Activation (Du-Lee 2018)
2. ReLU networks via landscape design (GLM18)
3. Matrix Completion (GLM16, GJZ17, . . .)
4. Rank $k$ approximation (Baldi-Hornik 89)
5. Matrix Sensing (BNS16)
6. Phase Retrieval (SQW16)
7. Orthogonal Tensor Decomposition (AGHKT12, GHJY15)
8. Dictionary Learning (SQW15)
9. Max-cut via Burer Monteiro (BBV16, Montanari 16)
Goal: Design the Loss Function so that gradient decent finds good solutions (e.g. no spurious local minimizers)\(^a\).

\(^a\) Janzamin-Anandkumar, Ge-Lee-Ma, Du-Lee

**Figure:** Illustration: SGD succeeds on the right loss function, but fails on the left in finding global minima.
Practical Landscape Design - Overparametrization

(a) Original Landscape  
(b) Overparametrized Landscape

Figure: Data is generated from network with $k_0 = 50$ neurons. Overparametrized network has $k = 100$ neurons\(^1\).

Without some modification of the loss, SGD will get trapped.

\(^1\) Experiment was suggested by Livni et al. 2014
If SGD is not finding a low training error solution, then fit a more expressive model until the training error is near zero.

How much over-parametrization do we need to efficiently optimize and generalize?

- Adding parameters increases computational and memory cost.
- Too many parameters may lead to overfitting (???).
Motivating Question

How much overparametrization ensures success of SGD?

- Empirically $p \gg n$ is necessary, where $p$ is the number of parameters.
- Very unrigorous calculations suggest $p = \text{constant} \times n$ suffices.
Deep Feedforward Networks

\[ x^{(0)} = \text{input data} \]
\[ x^{(l)} = \sigma(W_l x^{(l-1)}) \]
\[ f(x) = a^\top x^{(L)} \]
Interlude: Residual Networks

Deep Feedforward Networks

\[ x^{(0)} = \text{input data} \]
\[ x^{(l)} = \sigma(W_l x^{(l-1)}) \]
\[ f(x) = a^\top x^{(L)} \]

Empirically, it is difficult to train deep feedforward networks so Residual Networks were proposed:
Interlude: Residual Networks

Deep Feedforward Networks

\[ x^{(0)} = \text{input data} \]
\[ x^{(l)} = \sigma(W_l x^{(l-1)}) \]
\[ f(x) = a^\top x^{(L)} \]

Empirically, it is difficult to train deep feedforward networks so Residual Networks were proposed:

Residual Networks (He et al.)

ResNet of width \( m \) and depth \( L \):

\[ x^{(0)} = \text{input data} \]
\[ x^{(l)} = x^{(l-1)} + \sigma(W_l x^{(l-1)}) \]
\[ f(x) = a^\top x^{(L)} \]
Consider a width $m$ and depth $L$ residual network with a smooth ReLU activation $\sigma$ (or any differentiable activation). Assume that $m = O(n^4L^2)$, then gradient descent converges to a global minimizer with train loss 0.

- Same conclusion for ReLU, SGD, and variety of losses (hinge, logistic) if $m = O(n^{30}L^{30})$ (see Allen-Zhu-Li-Song and Zou et al.)
Two layer net: \( f(x) = \sum_{r=1}^{m} a_r \sigma(w_r^\top x) \).

How much do parameters need to move?

- Assume \( a_r^0 = \pm \frac{1}{\sqrt{m}} \), \( w_r^0 \sim N(0, I) \), and \( \|x\| = 1 \).
- Let \( w_r = w_r^0 + \delta_r \). Crucial Lemma: \( \delta_r = O\left(\frac{1}{\sqrt{m}}\right) \) moves the prediction by \( O(1) \).
Two layer net: $f(x) = \sum_{r=1}^{m} a_r \sigma (w_r^\top x)$.

**How much do parameters need to move?**

- Assume $a_r^0 = \pm \frac{1}{\sqrt{m}}$, $w_r^0 \sim N(0, I)$, and $\|x\| = 1$.
- Let $w_r = w_r^0 + \delta_r$. Crucial Lemma: $\delta_r = O\left(\frac{1}{\sqrt{m}}\right)$ moves the prediction by $O(1)$.

As the network gets wider, then each parameter moves less, and there is a global minimizer near the random initialization.
Remarks

- Gradient Descent converges to global minimizers of the train loss when networks are sufficiently overparametrized.
- Current bound requires $n^4 L^2$ and in practice $n$ is sufficient.
- No longer true if the weights are regularized.
- The best generalization bound one can prove using this technique matches a kernel method\(^2\) (Arora et al., Jacot et al., Chizat-Bach, Allen-Zhu et al.).

\(^2\)includes low-degree polynomials and activations with power series coefficients that decay geometrically.
1. Training data \((x_i, y_i)\) with label \(y \in \{-1, 1\}\).

2. Classifier is \(\text{sign}(f(W; x))\), where \(f\) is a neural net with parameters \(W\).

3. Margin \(\bar{\gamma} = \min_i y_i f(W; x)\).

4. We assume networks are overparametrized and can separate the data.
Margin Theory

Normalized margin \( \gamma(W) = \min_i y_i f(\frac{W}{\|W\|_2}, x_i) \). When \( \gamma \) is large, the network predicts the correct label with high confidence.

- Large margin guarantees generalization bounds (Bartlett et al., Neyshabur et al., Golowich et al.)

\[
\Pr(yf(W; x) < 0) \leq \frac{\mathcal{R}(W)}{\tilde{\gamma}}.
\]
Generalization via Margin Theory

Margin Theory

Normalized margin $\gamma(W) = \min_i y_i f\left(\frac{W}{\|W\|_2}, x_i\right)$. When $\gamma$ is large, the network predicts the correct label with high confidence.

- Large margin guarantees generalization bounds (Bartlett et al., Neyshabur et al., Golowich et al.)

\[
\Pr(y f(W; x) < 0) \lesssim \frac{\mathcal{R}(W)}{\tilde{\gamma}}.
\]

Large margin

Do we obtain large margin classifiers in Deep Learning?
Neural networks are trained via minimizing the regularized cross-entropy loss:

\[ \ell(f(W; x)) + \lambda \|W\|. \]
Regularized Loss

Neural networks are train via minimizing the regularized cross-entropy loss:

$$\ell(f(W; x)) + \lambda \|W\|.$$

Theorem (Wei-Lee-Liu-Ma 2018)

Let $f$ be a positive homogeneous network and

$$\gamma^* = \max_{\|W\| \leq 1} \min_{i \in [n]} y_i f(W; x_i)$$

be the optimal normalized margin.

- Minimizing cross-entropy loss is max-margin: $\gamma(W\lambda) \to \gamma^*$.
- The optimal margin is an increasing function of network size.
- Choosing a small but fixed $\lambda$ leads to approximate max-margin.
- When $f(x) = \langle w, x \rangle$ reduces to the result of Rosset, Zhu, and Hastie.
Imagine $\lambda$ is very small, so that $y_i f(W; x_i)$ is very large.

$$L_\lambda(W) = \sum_i \log(1 + \exp(-y_i f(W; x_i))) + \lambda\|W\|$$

$$\approx \sum_i \exp(-y_i f(W; x_i)) + \lambda\|W\|$$

$$\approx \max_{i \in [n]} \exp(-y_i f(W; x_i)) + \lambda\|W\|$$

$$\approx \exp(-\gamma(W)) + \lambda\|W\|.$$

Thus among solutions with the same norm, we will obtain a solution with $\gamma(W)$ largest.
Does large margin lead to parameter-independent generalization in Neural Networks?
Does large margin lead to parameter-independent generalization in Neural Networks?

Parameter-independent Generalization Bounds (Neyshabur et al.)

Let $f(W; x) = W_2 \sigma(W_1 x)$.

$$\Pr \left( yf(W; x) < 0 \right) \lesssim \frac{1}{\gamma \sqrt{n}}.$$ 

- Completely independent of the number of parameters.
Let \( f(W; x) = W_L \sigma(W_{L-1} \ldots W_2 \sigma(W_1 x)) \).

\[
\Pr \left( yf(W; x) < 0 \right) \lesssim \sqrt{L} \frac{\prod_{j=1}^{L} \|W_j\|_F}{\bar{\gamma} \sqrt{n}}
\]

and \( \bar{\gamma} \) is un-normalized margin.
Deep Feedforward Network (Golowich, Rakhlin and Shamir)

Let $f(W; x) = W_L \sigma(W_{L-1} \ldots W_2 \sigma(W_1 x))$.

$$\Pr \left( y f(W; x) < 0 \right) \lesssim \sqrt{L} \frac{\prod_{j=1}^{L} \|W_j\|_F}{\bar{\gamma} \sqrt{n}}$$

and $\bar{\gamma}$ is un-normalized margin.

- $\frac{\prod_{j=1}^{L} \|W_j\|_F}{\bar{\gamma}} = \gamma$ is the normalized margin.
- $\prod_{j=1}^{L} \|W_j\|_F = \frac{1}{L^{L/2}} \|\text{vec}(W_1, \ldots, W_L)\|_2^L = \frac{1}{L^{L/2}} \|W\|_2^L$ at a minimizer.

$l_2$-regularizer guarantees a “size-independent” bound.
Does GD Minimize Regularized Loss?

Training Loss

Let $f(x; W) = \sum_{r=1}^{m} a_r \sigma(\langle w_r, x \rangle)$ with $\sigma = \text{ReLU}$.

$$
\min_{W} \sum_{i} \ell(f(x_i; W), y_i) + \frac{\lambda}{2} \sum_{r=1}^{m} (a_r^2 + \|w_r\|_2^2).
$$

1. Imagine the network is infinitely wide $m \rightarrow \infty$, and we run gradient descent.

2. The density $\rho = \frac{1}{m} \sum_{j=1}^{m} \delta(a_j, w_j)$ is updated according to a Wasserstein flow induced by gradient descent.
Theorem (Very Informal, see arXiv)

For a two-layer network that is infinitely wide (or \( \exp(d) \) wide), gradient descent with noise converges to a global minimum of the regularized training loss in number of iterations \( T \lesssim \frac{d^2}{\epsilon^4} \).

- Overparametrization helps gradient descent find solutions of low train loss\(^3\)
- Noise is crucial to minimize the regularized loss. The noise is not on the parameters \( w \), but on the density \( \rho \).

\(^3\)see also Chizat-Bach, Mei-Montanari-Nguyen

Jason Lee
Better Result for Quadratic Activation

Corollary

Let \( \sigma(z) = z^2 \). If \( m \geq \sqrt{n} \), then SGD finds a global minimum of the regularized loss. Furthermore if \( y \sum_{j=1}^{m_0} a_j \sigma(w_j^\top x) \geq 1 \). Then for \( n \gtrsim \frac{d m_0^2}{\epsilon^2} \), SGD finds a solution

\[
L_{te}(W_t) \lesssim \epsilon.
\]

The sample complexity is independent of \( m \), the number of neurons.
$p \gg n$, no regularization, no early stopping, and yet we do not overfit.

In fact, test error decreases even after the train error is zero.

Weight decay helps a little bit ($< 2\%$), but generalization is already good without any regularization.

Figure: Credit: Neyshabur et al. See also Zhang et al.
Experiment

Problem

Why does SGD (with no regularization) not overfit?

Figure: Credit: Neyshabur et al. See also Zhang et al.
Theorem

Let $f_i(W) \triangleq f(W; x_i)$ be the prediction of a differentiable homogeneous network on datapoint $x_i$. Gradient Descent converges\(^a\) to a first-order optimal point of the non-linear SVM:

$$\min \|W\|_2$$

$$\text{st } y_i f_i(W) \geq 1.$$  

**GD is implicitly regularizing \(\ell_2\)-norm of parameters.**

\(^a\)Technical assumptions on limits existing is needed.

Open Problem

Under what assumptions will GD converge to a global max-margin?
1. Quadratic Activation Network\(^4\): \(p(W) = WW^T\) leads to an implicit nuclear norm regularizer, and thus a preference for networks with a small number of neurons.

2. Linear Network\(^5\): \(p(W) = W_L \ldots W_1\) leads to a Schatten quasi-norm regularizer \(\|p(W)\|_{2/L}\).

3. Linear Convolutional Network: Sparsity regularizer \(\|\cdot\|_{2/L}\) in the Fourier domain.

4. Feedforward Network: Size-independent complexity bound\(^6\)

---

\(^4\)see also Gunasekar et al. 2017, Li et al. 2017

\(^5\)see also Ji-Telgarsky

\(^6\)Golowich-Rakhlin-Shamir
Conclusion and Future Work

1. Overparametrization: Designs the landscape to make gradient methods succeed.
   - Current theoretical results are off by an order of magnitude in the necessary size.

2. Generalization is possible in the over-parametrized regime.
   - Explicit Regularization: Leads to large margin classifiers, and low statistical complexity.
   - Implicit Regularization: The choice of algorithm and parametrization constrain the effective complexity of the chosen model.

3. We understand only very simple models and settings.
   - Deep Learning is used in a black-box fashion in many downstream tasks (e.g. as a function approximator)
References


2. Davis, Drusvyatskiy, Sham Kakade, and Jason D. Lee, *Stochastic subgradient method converges on tame functions.*


8. Du and Lee, *On the Power of Over-parametrization in Neural Networks with Quadratic Activation*
Thank You.

Questions?