A Convergence Theory for Deep Learning via Over-Parameterization

Zeyuan Allen-Zhu

MSR AI

Yuanzhi Li

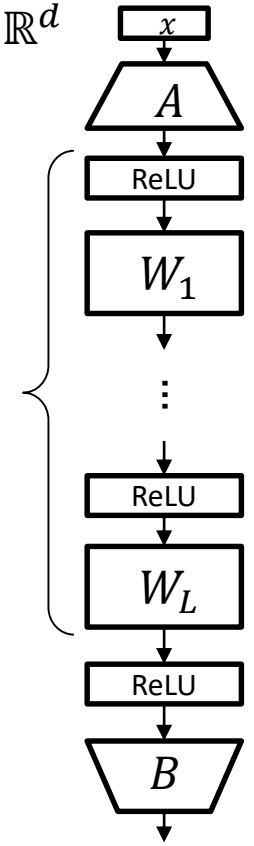
Stanford
Princeton

Zhao Song

UT Austin
U of Washington
Harvard

Main Result

samples $x_1, \ldots, x_n \in \mathbb{R}^d$



L hidden layers
$W_\ell \in \mathbb{R}^{m \times m}$

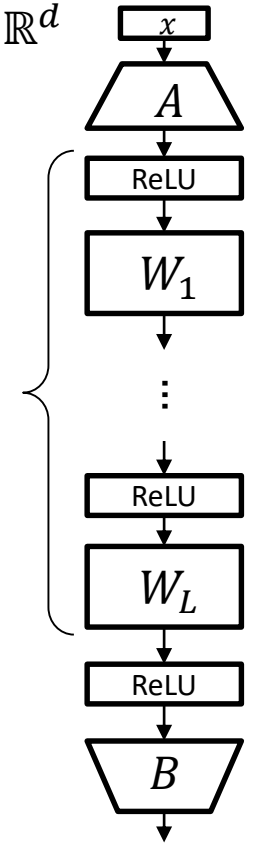
Main Theorem

If data non-degenerate (e.g. norm 1 and $\|x_i - x_j\|_2 \geq \delta$)
If overparameterized $m \geq poly(n, L, \delta^{-1})$

The main result is the following. Consider training L hidden layers of a deep neural network, given n training data points that are non-degenerate, meaning their pairwise relative distance is at least δ. Suppose the network is overparameterized, meaning the number of neurons is polynomial in n, L and δ^{-1}.

Main Result

samples $x_1, \dots, x_n \in \mathbb{R}^d$



L hidden layers
$W_\ell \in \mathbb{R}^{m \times m}$

Main Theorem

If data non-degenerate (e.g. norm 1 and $\|x_i - x_j\|_2 \geq \delta$)
If overparameterized $m \geq poly(n, L, \delta^{-1})$

Then, SGD finds training global minima in

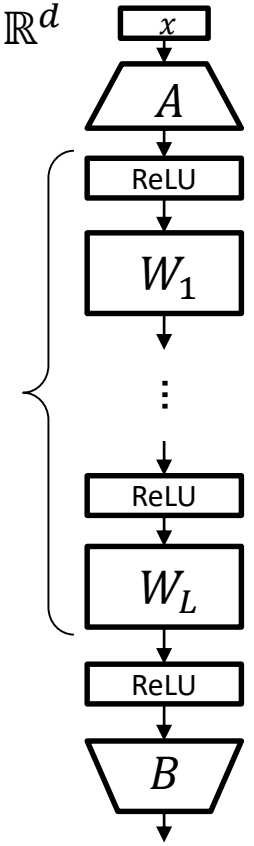
$$T = \frac{poly(n, L)}{\delta^2} \cdot \log \frac{1}{\varepsilon}$$

iterations for ℓ_2-regression

Then, we proved stochastic gradient descent can find global minima in polynomial time by training only hidden layers.

Main Result

samples $x_1, \dots, x_n \in \mathbb{R}^d$



L hidden layers
$W_\ell \in \mathbb{R}^{m \times m}$

Main Theorem

If data non-degenerate (e.g. norm 1 and $\|x_i - x_j\|_2 \geq \delta$)
If overparameterized $m \geq poly(n, L, \delta^{-1})$

Then, SGD finds training global minima in

$$T = \frac{poly(n, L)}{\delta^2} \cdot \log \frac{1}{\varepsilon}$$

iterations for ℓ_2-regression

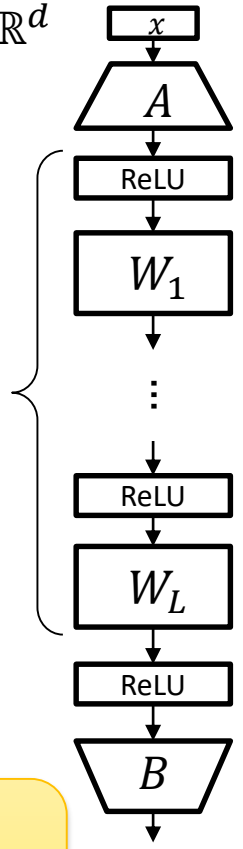
In paper:

- also for other smooth losses (cross-entropy, etc)
- also for other architectures (ResNet, CNN, etc)

Similar results also hold for other losses and other network architectures such as ResNet and CNN.
These can be found in the paper.

Key Message 1

samples $x_1, \dots, x_n \in \mathbb{R}^d$



Main Theorem

If data non-degenerate (e.g. norm 1 and $\|x_i - x_j\|_2 \geq \delta$)
If overparameterized $m \geq poly(n, L, \delta^{-1})$

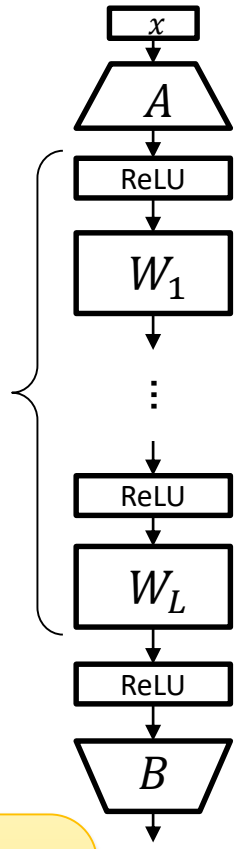
Then, SGD finds training global minima in

$$T = \frac{poly(n, L)}{\delta^2} \cdot \log \frac{1}{\varepsilon}$$

iterations for ℓ_2-regression

Our first key message is the following. Our theorem is obtained by training with respect to hidden layers, where prior work [Daniely, NeurIPS 2017] studies training essentially only the last layer, which is an easy convex problem.

Key Message 2: poly(L)



Main Theorem

If data non-degenerate (e.g. norm 1 and $\|x_i - x_j\|_2 \geq \delta$)
If overparameterized $m \geq poly(n, L, \delta^{-1})$

Then, SGD finds training global minima in

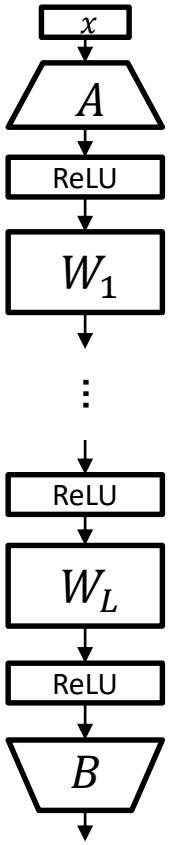
$$T = \frac{poly(n, L)}{\delta^2} \cdot \log \frac{1}{\varepsilon}$$

iterations for ℓ_2-regressio

Our second key message is the following. We prove polynomial dependence on the depth L. In contrast,

- The independent work [Du et al. ICML 19] needs exponential time in L
- Prior work [Daniely, NeurIPS 17] for training last layer also needs $e^{O(L)}$

Key Message 2: poly(L)



Main Theorem

If data non-degenerate (e.g. norm 1 and $\|x_i - x_j\|_2 \geq \delta$)
If overparameterized $m \geq poly(n, L, \delta^{-1})$

Then, SGD finds training global minima in

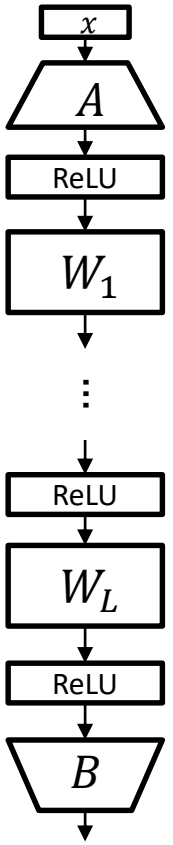
$$T = \frac{poly(n, L)}{\delta^2} \cdot \log \frac{1}{\varepsilon}$$

iterations for ℓ_2-regression

Intrinsically, our polynomial bound is possible because ReLU prevents exponential gradient explosion/vanishing, in a provable sense!

(for a sufficiently large region near random initialization)

Key Message 2: poly(L)



Main Theorem

If data non-degenerate (e.g. norm 1 and $\|x_i - x_j\|_2 \geq \delta$)
If overparameterized $m \geq poly(n, L, \delta^{-1})$

Then, SGD finds training global minima in

$$T = \frac{poly(n, L)}{\delta^2} \cdot \log \frac{1}{\varepsilon}$$

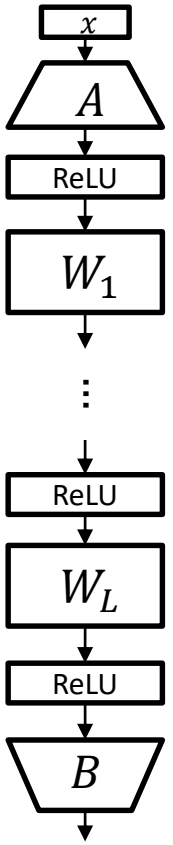
iterations for ℓ_2-regression

Intrinsically, our polynomial bound is possible because ReLU prevents exponential gradient explosion/vanishing, in a provable sense!

(for a sufficiently large region near random initialization)

In contrast, getting $e^{O(L)}$ is almost trivial: each hidden weight matrix W_ℓ has spectral norm 2, so overall 2^L. The hard part is proving $poly(L)$.

Key Message 3: almost-convex geometry



The third key message is the following. We prove in the paper, for a sufficiently large neighborhood of the random initialization, the training objective is ***almost convex***.

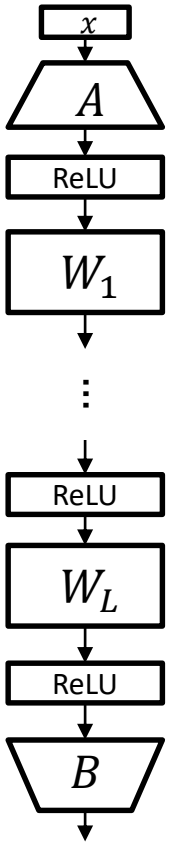
Key Message 3: almost-convex geometry

Main Lemma

If loss is large, then gradient is large:

$$\left\|\nabla F\left(\vec{W}\right)\right\|_F^2 \geq F\left(\vec{W}\right) \cdot \left(\delta/n^2\right)$$

(after appropriate normalization)



This means, if the objective is large, then gradient is large.

Key Message 3: almost-convex geometry

Main Lemma

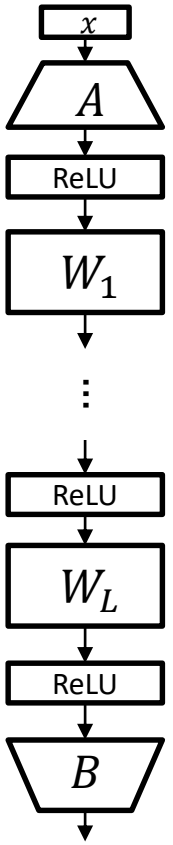
If loss is large, then gradient is large:

$$\left\|\nabla F\left(\vec{W}\right)\right\|_F^2 \geq F\left(\vec{W}\right)\cdot(\delta/n^2)$$

Main Lemma

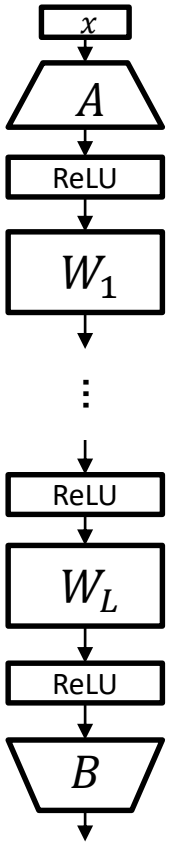
Objective is semi-smooth:

$$F\left(\vec{W}+\vec{W'}\right)=F\left(\vec{W}\right)+\left\langle\nabla F\left(\vec{W}\right),\vec{W'}\right\rangle\pm poly(n,L)\cdot\left\|\vec{W'}\right\|_F$$



Also, the objective is sufficiently smooth, meaning that if you move in the negative gradient direction, the objective value can be sufficiently decreased.

Key Message 3: almost-convex geometry



Main Lemma

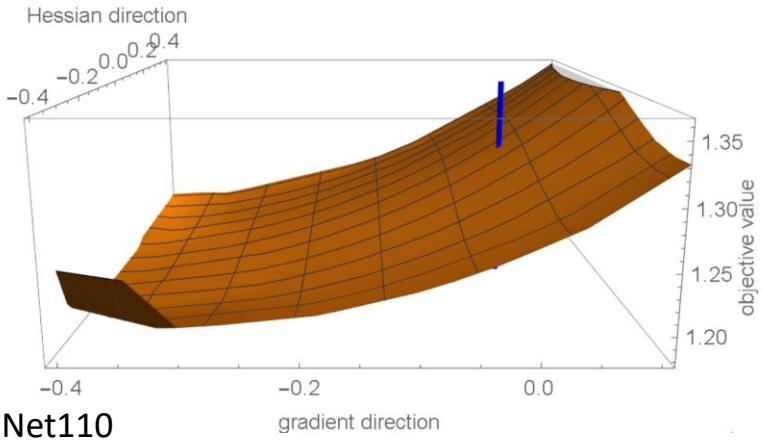
If loss is large, then gradient is large:

$$\left\|\nabla F\left(\vec{W}\right)\right\|_F^2 \geq F\left(\vec{W}\right)\cdot\left(\delta/n^2\right)$$

Main Lemma

Objective is semi-smooth:

$$F\left(\vec{W}+\vec{W'}\right)=F\left(\vec{W}\right)+\left\langle\nabla F\left(\vec{W}\right),\vec{W'}\right\rangle\pm poly(n,L)\cdot\left\|\vec{W'}\right\|_F$$



CIFAR10/100
VGG19/ResNet32/ResNet110

We verified this is true also on real data. Goodfellow et al. [ICLR 2015] also observed this phenomenon but a proof was not known.

Key Message 3: almost-convex geometry

Main Lemma

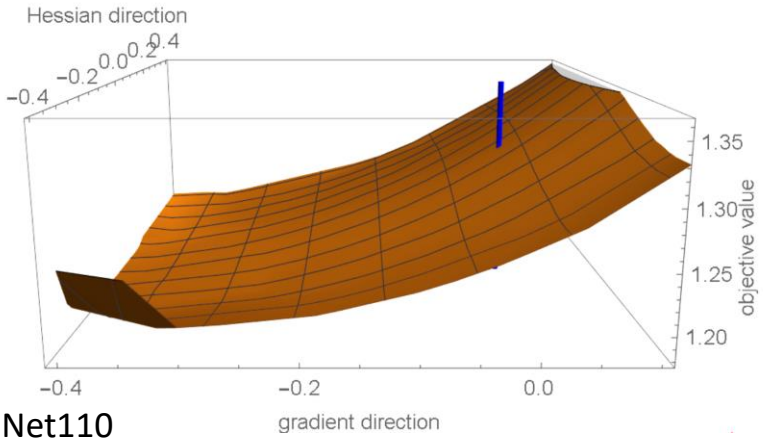
If loss is large, then gradient is large:

$$\left\|\nabla F\left(\vec{W}\right)\right\|_F^2 \geq F\left(\vec{W}\right)\cdot\left(\delta/n^2\right)$$

Main Lemma

Objective is semi-smooth:

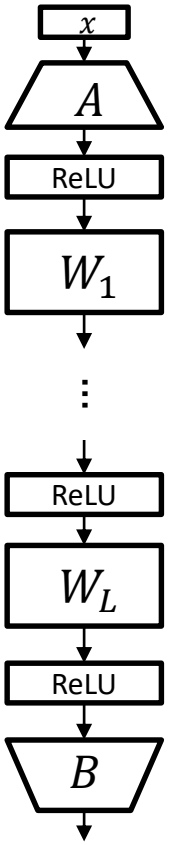
$$F\left(\vec{W}+\vec{W'}\right)=F\left(\vec{W}\right)+\left\langle\nabla F\left(\vec{W}\right),\vec{W'}\right\rangle\pm poly(n,L)\cdot\left\|\vec{W'}\right\|_F$$



CIFAR10/100
VGG19/ResNet32/ResNet110

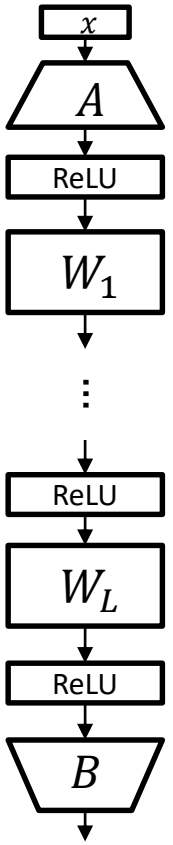
Main Theorem

SGD finds global minima in polynomial time



These two main lemmas together imply our main theorem.

Equivalent View: neural tangent kernel

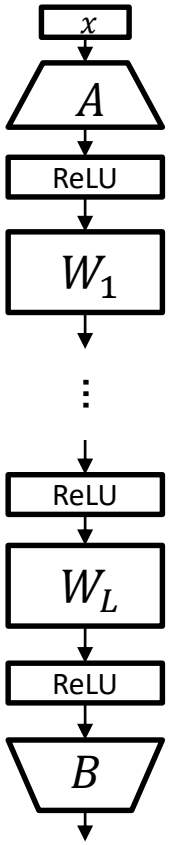


In fact… we proved

If $m \geq poly(n, L)$, for a sufficiently large neighborhood of the random initialization, neural networks behave like Neural Tangent Kernel (NTK).

Finally, let us take an alternative view.
If one goes into the paper, we proved the following. If m, the number of neurons, is polynomially large, then for a sufficiently large neighborhood of the random initialization, neural networks behave nearly identical to the so-called neural tangent kernels, or NTK.

Equivalent View: neural tangent kernel



In fact… we proved

If $m \geq poly(n, L)$, for a sufficiently large neighborhood of the random initialization, neural networks behave like Neural Tangent Kernel (NTK).

- $\nabla F\left(\vec{W}\right) = \left(1 \pm \frac{1}{\sqrt{m}}\right) \cdot$ feature space of NTK
- $F\left(\vec{W}^*\right) = F^{NTK}\left(\vec{W}^*\right) \pm \frac{1}{m^{1/6}}$

Specifically, this means two things. The gradient behaves like NTK, and the objective behaves like NTK.

Conclusion

We proved

If $m \geq poly(n, L)$, within certain initialization and learning rate regime,

Over-parameterized deep networks = Neural Tangent Kernel (NTK).

$\Longrightarrow$ networks essentially convex and smooth $\Longrightarrow$ training is EASY

In other words, we proved that within certain parameter regime, over-parameterized deep neural networks behave nearly the same as NTK. Therefore, the training task is essentially convex, so training is easy.

Conclusion

We proved

If $m \geq poly(n, L)$, within certain initialization and learning rate regime,

Over-parameterized deep networks = Neural Tangent Kernel (NTK).

$\Longrightarrow$ networks essentially convex and smooth $\Longrightarrow$ training is EASY

*Author Note: for other regimes, neural networks provably **more powerful** than NTK*

See [A-L, 1905.10337], "What Can ResNet Learn Efficiently, Going Beyond Kernels?"

Note this is **not true** for other learning rate regimes, and neural networks can be provably more powerful than NTK, see our follow-up work.

Conclusion

We emphasize again that, prior work studying the relationship to NTK either requires $m = \infty$ or $m \geq e^{\Omega(L)}$. Our result is **polynomial in the depth L**.

We proved

If $m \geq poly(n, L)$, within certain initialization and learning rate regime,

Over-parameterized deep networks $=$ Neural Tangent Kernel (NTK).

$\Longrightarrow$ networks essentially convex and smooth $\Longrightarrow$ training is EASY

Conclusion

We emphasize again that, prior work studying the relationship to NTK either requires $m = \infty$ or $m \geq e^{\Omega(L)}$. Our result is **polynomial in the depth L**.

We proved

If $m \geq poly(n, L)$, within certain initialization and learning rate regime,

Over-parameterized deep networks = Neural Tangent Kernel (NTK).

$\Longrightarrow$ networks essentially convex and smooth $\Longrightarrow$ training is EASY

Thanks!