Convergence Analysis of Stochastic Gradient Descent on Overparametrized Neural Networks

Eric Zhang, Wenhan Yang
Advised by Prof. Shuyang Ling

October 30, 2021

Agenda

- Preliminary
- Motivation and Related Work
- Experiment and Generalization
- Reflection and Future Work

Preliminary

Machine Learning: Given dataset (x_i, y_i) where $1 \leq i \leq n$, find a best-fit function f among a family of functions and use it to predict y^* given any new data x^* (E.g. linear regression)

Neural Network: a family of nonlinear functions determined by its depth, width, and activation function.

How does NN work: The inputs go through matrix multiplication and non-linear activation function.

Find best-fit function: minimizing the loss function $L(w_1, w_2, ...)$ where w_i are weights of each layer. E.g. MSEloss.

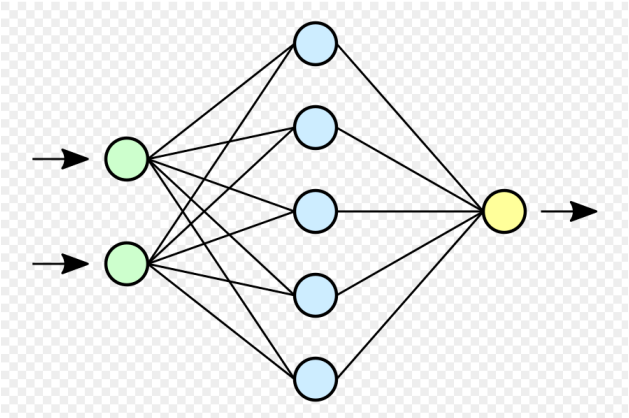


Figure: A 2-layer neural network

Preliminary

Gradient Descent:

$$W(t+1) = W(t) - \eta \nabla \frac{\partial L(W(t))}{\partial W(t)}$$

Stochastic Gradient Descent: When calculating the MSE loss, use only a random batch of samples instead of all samples.

Preliminary

Overparametrized:

$$m \geq n$$

where m is number of nodes in hidden layers and n is number of data

Motivation and Related Work

[Du, 2019]: For an m hidden node, two-layer NN with Relu activation function, as m is big enough, randomly initialized GD can achieve zero training loss at a linear convergence rate for quadratic loss function, where

$$Relu(x) = max(0, x)$$

This is surprising because the objective function is non-convex and even non-smooth. If we run GD, it may stuck at saddle points or local minimum.

Motivation and Related Work

Definition 1 Two-layer neural network:

$$f(\mathbf{W}, \mathbf{a}, \mathbf{x}) = \frac{1}{\sqrt{m}} \sum_{r=1}^{m} a_r \sigma(w_r^\top \mathbf{x})$$

where $\mathbf{x} \in \mathbb{R}^d$ is the input, $w_r \in \mathbb{R}^d$ is the weight vector of the first layer, $a_r \in \mathbb{R}$ is the weight of the second layer, and $\sigma(\cdot)$ is the Relu.

Definition 2 Loss function:

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

Motivation and Related Work

Definition 3 Prediction vector $> \mathbf{u}(t) := (u_1(t), ..., u_n(t)) \in \mathbb{R}^n$ where $u_i(t) = f(\mathbf{W}(t), \mathbf{a}, \mathbf{x}(\mathbf{i}))$ is the prediction on input x_i at time t.

Definition 4 (Key definition) $H^\infty \in \mathbb{R}^{n \times n}$ where

$$H_{ij}^\infty = \mathbb{E}_{\mathbf{W} \sim N(0,I)}[x_i^\intercal x_j \mathbb{I}\{w^\intercal x_i \geq 0, w^\intercal x_j \geq 0\}]$$

and we denote

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

If we assume no parallel inputs, then $\lambda_0 > 0$

Motivation and Related Work

Theorem 1 (Convergence Rate of GD). Suppose $||x_i||_2 = 1$ and $|y_i| \leq C$ for some constant C, then if we set the number of hidden nodes $m = \Omega(\frac{n^6}{\lambda_0^4 \delta^3})$ and we i.i.d. initialize $w_r \sim \mathbb{N}(0, I)$, $a_r \sim unif[\{-1, 1\}]$ for $r \in [m]$, and step size $\eta = O(\frac{\lambda_0}{n^2})$, then with probability at least $1 - \delta$, we have:

$$||u(k) - y||_2^2 \leq (1 - \frac{\eta \lambda_0}{2})^k ||u(0) - y||_2^2$$

Motivation and Related Work

Theorem 1 is proved by the following key fact:

$$y - u(t+1) = (I - \eta H(t))(y - u(t))$$

where $H(t) \in \mathbb{R}^{n \times n}$ and

$$H(t)_{ij} = \frac{1}{m} x_i^\top x_j \sum_{r=1}^{m} \mathbb{I}\{x_i^\top w_r(t) \geq 0, x_j^\top w_r(t) \geq 0\}$$

The key observation is: Though $H(t)$ is changing with respect to t, we always have $||H(t) - H(0)||_2 = O(\sqrt{\frac{1}{m}})$ and $||H(0) - H^\infty||_2 = O(\sqrt{\frac{1}{m}})$. Thus, the predictions are completed captured by H^∞

Experiment and Generalization

We make generalizations into stochastic settings and tried the following:

- From 2 layers to multi layers
- From Relu to other acitivation functions

Also, we tried to see whether we can find an improved bound for m.

To start, we sampled (x_i, y_i) for $1 \leq i \leq 20$ where

- $x_i \in \mathbb{R}^{100}$
- $x_{ik} \sim unif(-1, 1)$ and $||x_i||_2 = 1$ for $1 \leq k \leq 100$
- $y_i \sim \mathbb{N}(0, 1)$

SGD on 2 Layers

We mimic the initialization condition in Theorem 1: $w_r \sim \mathbb{N}(0, I)$, $a_r \sim unif[\{-1, 1\}]$, then we set the number of hidden nodes m to be $[10, 20, 100, 500, 1000, 5000]$

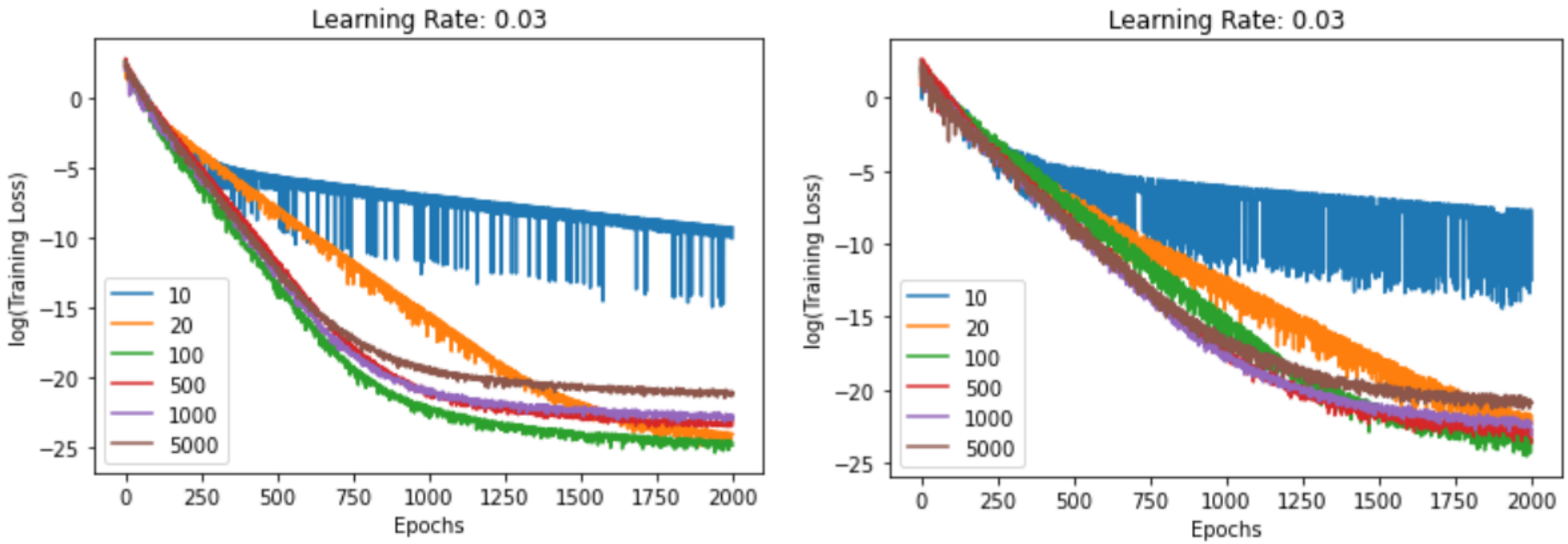
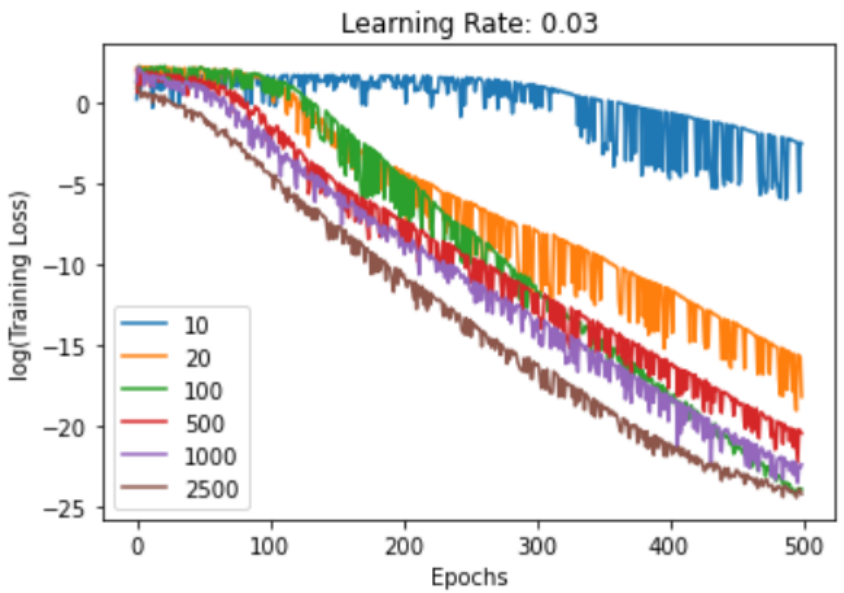
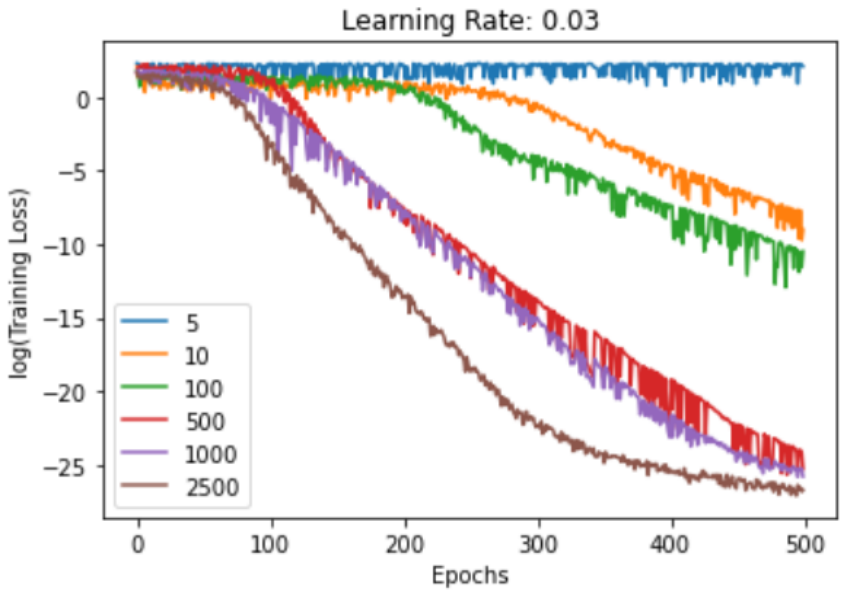


Figure: (a) Batch size = 15 (b) Batch size = 10

Observation: global convergence but fluctuating speed as hidden layer nodes increase.

SGD on Multi Layers





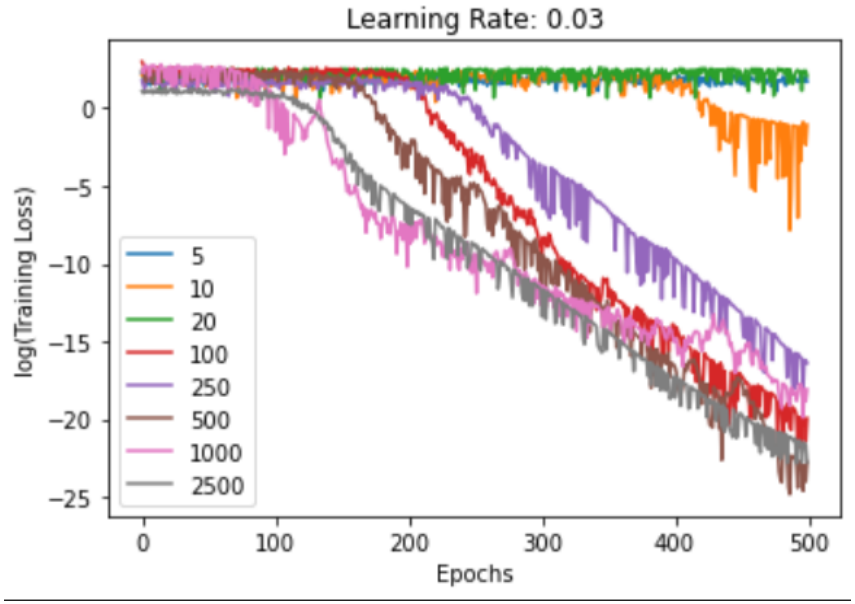


Figure: 3,4,5 layers

SGD on Different Activation Functions

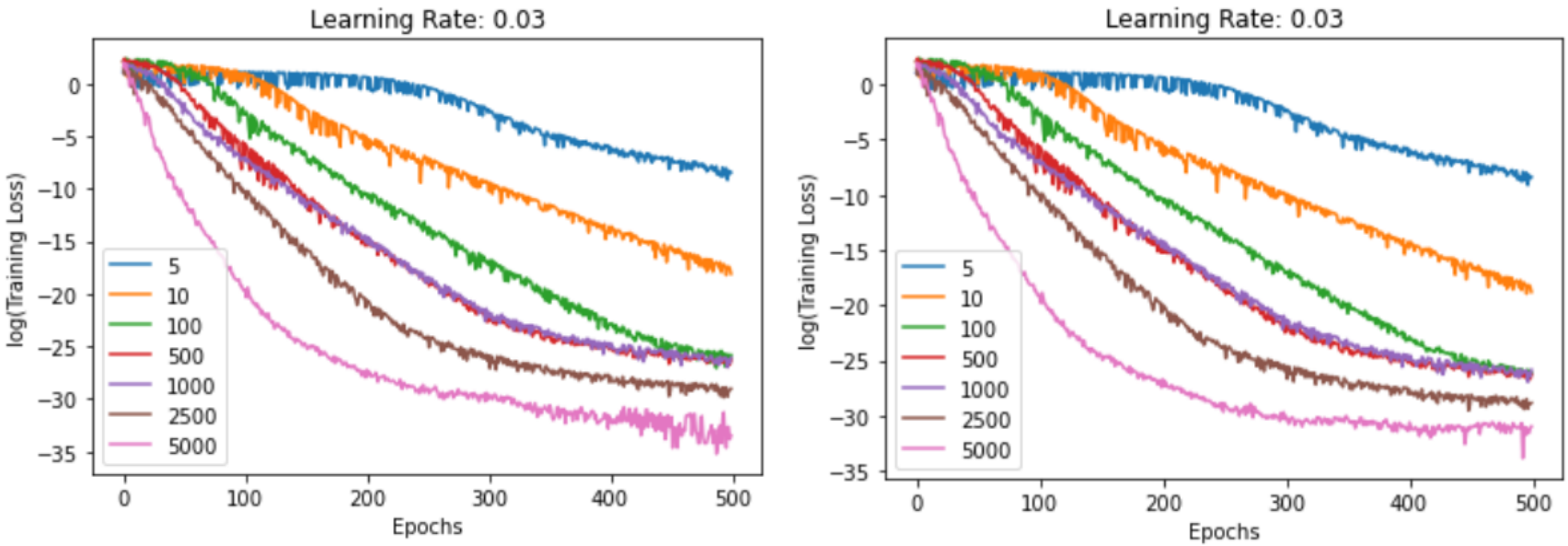


Figure: 3-layer NN with both tanh and tanh+relu

Recall H^{∞} is the key and it's induced by Relu. Here, we may need to induce another matrix and use similar techniques to prove the result.

How Much Overparametrization is Needed?

Motivation: Unrealistic order of m if n is large. Recall that $m = \Omega(\frac{n^6}{\lambda_0^4 \delta^3})$ in Du's paper.

We conjecture that if $m \times h > n$, then GD will find the global optimal (exact overparametrized). Intuition: Polynomial Interpolation.

Conjecture Empirical Validation

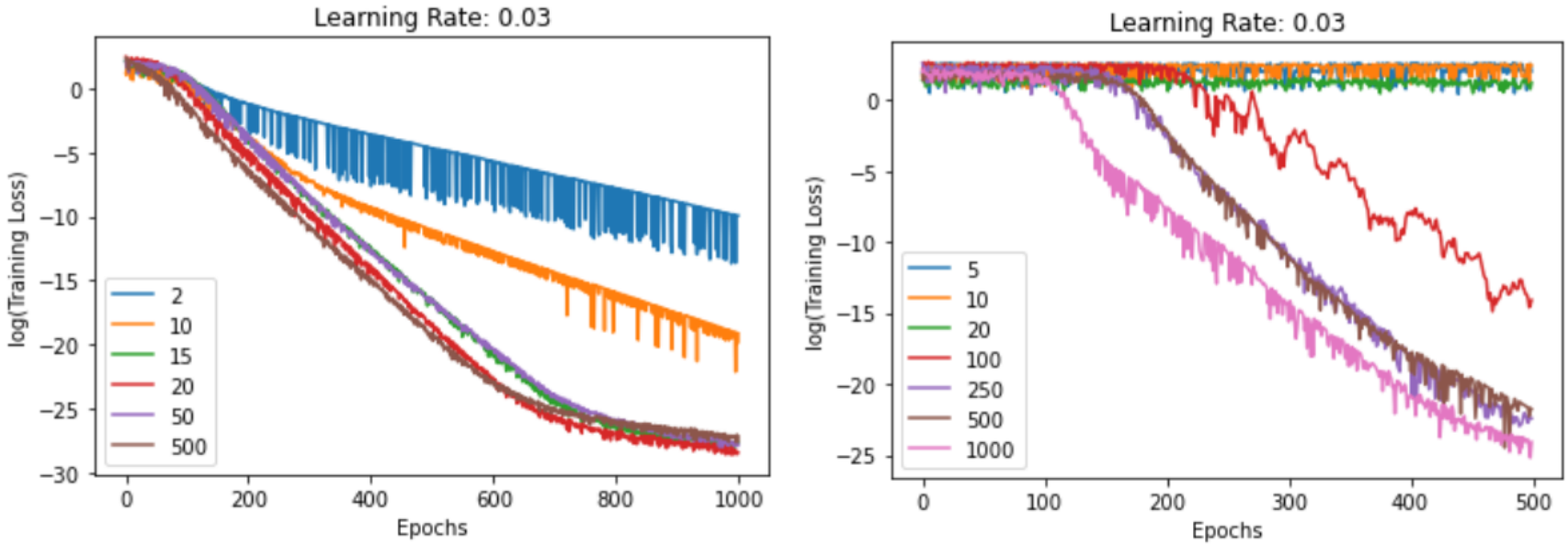


Figure: 2-layer with n = 10; 3-layer with n = 50

Reflection and Future Work

- Generalizing to multi-layers and different activation functions are feasible.
- We can improve the bound for m but more theoretical work is need (Use more advanced concentration inequalities in some steps of proofs)

References

[1] S. S. Du, X. Zhai, B. Poczos, and A. Singh, "Gradient descent provably optimizes over-parameterized neural networks," 2019.

[2] S. S. Du, J. D. Lee, H. Li, L. Wang, and X. Zhai, "Gradient descent finds global minima of deep neural networks," 2019.

[3] Z. Allen-Zhu, Y. Li, and Z. Song, "A convergence theory for deep learning via over-parameterization," 2019.