technische universität
dortmund

SFB 876 Providing Information
by Resource-Constrained Data Analysis

Towards Optimal Bounds on the Width of Neural Networks

joint work with A. Munteanu (TU Dortmund), Z. Song (Adobe Research) and D. Woodruff (CMU)

Simon Omlor | DoDSc 2021

Motivation

- Neural networks have been a popular topic for recent research;
- Even though they perform well in practice little is known about theoretical bounds

Motivation

- Neural networks have been a popular topic for recent research;
- Even though they perform well in practice little is known about theoretical bounds

Our goal: Analyze worst case behavior of ’simple’ neural networks.

Two layer ReLU network

Assume that our data points are points in $\mathbb{R}^d$. Then a two layer ReLU network is given by:

- weights of the first layer, i.e. $w_1 \dots w_m \in \mathbb{R}^d$;
- weight vector $a \in \{-1, 1\}^m$ for the second layer;

We set $f(W, x, a) := \sum_{j=1}^{m} a_j \mathrm{ReLU}(\langle w_j, x \rangle)$ (where $\mathrm{ReLU}(r) = \max\{r, 0\}$) to be the prediction of x.

Train a Two layer ReLU network

Assume that we are given a data set consisting of points $x_1 \dots, x_n \in \mathbb{R}^d$ together with predictions $y_1, \dots, y_n \in \mathbb{R}$. In order to get good predictions one tries to optimize

$$R(W, X) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(W, x_i, a), y_i)$$

where $\ell : \mathbb{R}^2 \to \mathbb{R}_+$ is an appropriate loss function.

Train a Two layer ReLU network

Assume that we are given a data set consisting of points $x_1 \dots, x_n \in \mathbb{R}^d$ together with predictions $y_1, \dots, y_n \in \mathbb{R}$. In order to get good predictions one tries to optimize

$$R(W,X) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(W,x_i,a),y_i)$$

where $\ell : \mathbb{R}^2 \to \mathbb{R}_+$ is an appropriate loss function. Our loss functions:

$\ell_1(f(W,x_i,a),y_i) = \ln(1+\exp(y_i \cdot f(W,x_i,a)))$	logistic loss; $y_i \in \{-1,1\}$
$\ell_2(f(W,x_i,a),y_i) = (f(W,x_i,a) - y_i)^2$	squared loss; $y_i \in \mathbb{R}$

Our goal

Training error $R(W, X) \leq \epsilon$

- Number m of inner notes;
- Number of iterations needed.

Our results

References	Width m	Iterations T	Loss function
[Du et at. 2019]	$O(\lambda^{-4}n^6)$	$O(\lambda^{-2}n^2\log(1/\epsilon))$	squared loss
[Song, Yang 2019]	$O(\lambda^{-4}n^4)$	$O(\lambda^{-2}n^2\log(1/\varepsilon))$	squared loss
Our work	$O(\lambda^{-2}n^2)$	$O(\lambda^{-2}n^2\log(1/\varepsilon))$	squared loss
[Ji, Telgarsky 2020]	$O(\gamma^{-8}\log n)$	$\tilde{O}(\varepsilon^{-1}\gamma^{-2})$	logistic loss
Our work	$O(\gamma^{-2}\log n)$	$\tilde{O}(\varepsilon^{-1}\gamma^{-2})$	logistic loss
[Ji, Telgarsky 2020]	$\Omega(\gamma^{-1/2})$	N/A	logistic loss
Our work	$\Omega(\gamma^{-1}\log n)$	N/A	logistic loss

Summary of previous work and our work. The improvements are mainly in the dependence on the parameters λ, γ, n affecting the width m.

Initalization sheme

Paired initialization:

- For each $r = 2i - 1$, we choose w_r to be a random Gaussian vector drawn from $\mathcal{N}(0, I)$.
- For each $r = 2i - 1$, we choose $a_r = 1$.
- For each $r = 2i$, we choose $w_r = w_{r-1}$.
- For each $r = 2i$, we choose $a_r = -1$.

Initalization sheme

Paired initialization:

- For each $r = 2i - 1$, we choose w_r to be a random Gaussian vector drawn from $\mathcal{N}(0, I)$.
- For each $r = 2i - 1$, we choose $a_r = 1$.
- For each $r = 2i$, we choose $w_r = w_{r-1}$.
- For each $r = 2i$, we choose $a_r = -1$.

$\rightarrow$ Allows us to scale the vectors w_r as $f(W, x_i, a) = 0$ for all $i \in [n]$.

Gradient descent/NTK analysis

Update step:

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

Idea of the analysis:

$$\frac{\partial f(W, x, a)}{\partial w_r} = a_r x \mathbf{1}_{w_r^\top x \geq 0}$$

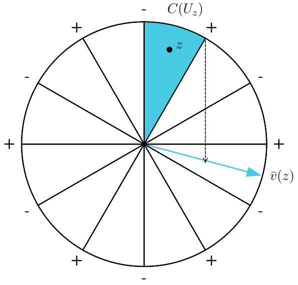
does not change by much over all iterations if m is large enough.

Lower bound

Consider the following example data set:

$$x_k = \left(\cos\left(\frac{2k\pi}{n}\right), \sin\left(\frac{2k\pi}{n}\right)\right)$$

$$y_k = (-1)^k$$



'Bad' example

Lower bound

Theorem 1

There exists a data set in 2-dimensional space, such that any two-layer ReLU neural network with width $m = o(\gamma^{-1})$ necessarily misclassifies at least $\Omega(n)$ points.

Outlook/Future work

- What is the worst case bounds of m for logistic loss: $\tilde{O}(\gamma^{-1})$ or $\tilde{O}(\gamma^{-2})$?
- What is the worst case bounds of m for squared loss: $\tilde{O}(n)$ or $\tilde{O}(n^2)$?.
- What bounds can be shown for networks with different activation functions/loss functions/more than two layers